POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

# FPGA-Assisted Verification of Digitally-Assisted Analog-to-Digital Converter Calibration Algorithms

Laurea Magistrale in Electronics Engineering - Ingegneria Elettronica

**Author:** Simone Guglielmino

**Advisor:** Prof. Andrea G. Bonfanti

**Co-advisor:** Gabriele Be'

**Academic year:** 2021-2022

## 1. Introduction

The quick growth in the complexity of Integrated Circuits (ICs) poses serious challenges to the ability to accurately verify the behavior of such complex systems.
This master thesis presents a new verification method that implements an 8-channel Time-Interleaved (TI) Analog-to-Digital Converter (ADC), which requires extensive digital calibrations, on a Field-Programmable Gate Array (FPGA). The focus of this *thesis* is the verification of the calibration algorithms of the converter, which are needed to compensate for the effects of gain and offset mismatch between the cores.
The implementation on the FPGA allows simulating in nearly real-time these calibration algorithms for hours/days in order to detect the presence of misbehaviors that can occur in such a long time, like, for instance, saturation or drift of the accumulators or integration of errors caused by the finite precision of digital implementation. These flaws of the algorithms are difficult to detect in standard VHDL or MATLAB simulations. In fact, VHDL simulations are too slow to detect these errors in a reasonable amount of time, whereas MATLAB is not suited to accurately and realistically describe digital circuits of reasonable complexity. To achieve this task we have also developed a digital approximation of the analog sections of the converter.

## 2. Verification of ICs

Verification is a fundamental step in the ICs design flow. It is defined as the process of evaluating if a system, in this case, an IC, complies with the specifications. More simply, verification attempts to answer the question *"does the circuit do what is intended for?"*.

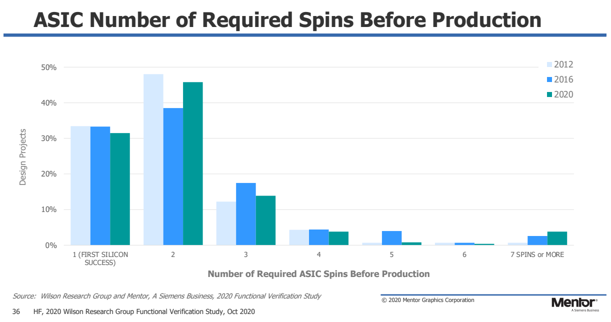

Figure 1: Number of *re-spins* required before production [1].

The importance of a correct and complete verification of the IC cannot be underestimated. Bugs undetected during verification require a new set of photolithographic masks in a so-called *re-spin*. This is a common occurrence in the semiconductor industries as less than 35% of the ASICs are able to reach the so-called *first silicon success*, as reported in the largest survey of ASIC/FPGA industries [1] and shown in figure 1. The possibility that an IC fails to behave as expected, coupled with the growing costs associated with a new set of photolithographic masks, gives the utmost importance to the verification phase. The study in [1] also shows how in the last decade the number of verification engineers has overcome the number of design engineers, especially for large designs (more than 80-$M$ equivalent gates) [1]. Nowadays, up to 70% of the engineering time is dedicated to verification.

Verification itself includes multiple different aspects and can be divided into *pre-silicon* and *post-silicon* verification. Pre-silicon verification acts before manufacturing the ICs to validate the design, whereas post-silicon one uses manufactured ICs (for instance, early production batches) to check the correct behavior of the device. We will focus on pre-silicon, and, more precisely, on functional verification, which is defined as the process of ensuring that the system behaves as intended. Logical or functional flaws represent, in fact, about 50% of the design flaws requiring a re-spin [1].

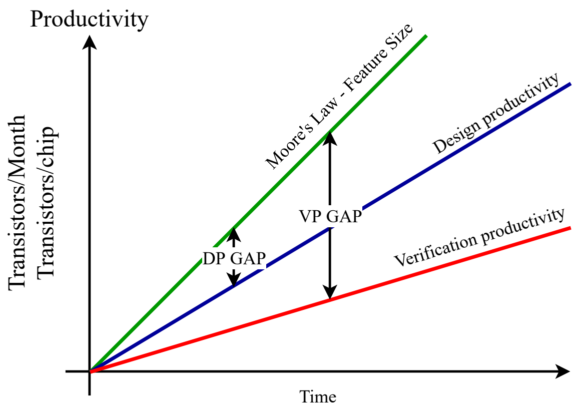

Figure 2: Evolution trend of manufacturing capability, design productivity, and verification productivity.

As predicted by Moore's Law, the number of transistors in a (digital) IC doubles approximately every 24 months. This continuous evolution has been made possible by the quick progress of digital Electronic Design Automation (EDA) tools, used both to design and validate the ICs. Digital verification is probably the electronic design automation sub-field that has undergone the fastest evolution in the last decades in order to close the verification-productivity (VP) gap as shown in figure 2 [2].

Going from custom-developed test benches (TB) to fully-automated test environments has been made possible by the use of Universal Verification Methodology (UVM). In the last few years, a new family of techniques, known as formal verification, has started gaining popularity. The idea is to use formal mathematics methods (theorem proving, induction, model checking, etc.) to prove the correct working of a system. Compared to standard RTL simulations, formal verification is equivalent to simulating a design in all possible ways, resulting in a 100% verification coverage.

However, the explosive growth of the complexity of the circuits still poses another issue. Even with computers getting more powerful every year, the time required to simulate, even at behavioral level, an entire IC keeps growing year after year, with similar effects on the time-to-market and design costs. For this reason, in the last decades, the concept of HW emulation and FPGA prototyping has become widespread in the semiconductor industries. HW emulation exploits custom ASICs to process an opportunely compiled version of the RTL code, whereas FPGA prototyping implements the RTL code on an FPGA to test it on real hardware, at close to real speeds. Both techniques allow simulating large designs in reasonable times.

FPGA prototyping is exactly the technique we propose as verification method to check the correct behavior of the digital calibration algorithms of the TI converter. However, since an ADC converter is a mixed-signal circuit, this process requires the implementation also of analog parts, like for example the generation of a digital signal that approximates with sufficient precision the analog one to be sampled by the A/D converter. Thus, for all intents and purposes, the proposed method is a "mixed-signal", rather than "only-digital", kind of verification.

## 3. Time-Interleaved Analog-to-Digital Converters

The first TI converter in literature has been presented in 1980 in a paper by W.C. Black and D.A. Hodges [3]. A TI-ADC is built by placing multiple sub-converters (cores, channels, or slices) in parallel, as shown in figure 3.
Each channel is driven by a different phase of the same clock signal, setting a different sampling instant for each of the cores. This, effectively, results in a conversion rate equal to $M$ times the conversion rate of the single sub-converter, with $M$ being the number of sub-converters placed in parallel, usually called interleaving factor.
TI converters allow increasing the conversion rate of the whole ADC while keeping the single sub-converters in their efficient working zone. This enables, for instance, the use of a common CMOS process instead of a more exotic one to implement a high-speed converter, with obvious advantages in terms of cost and integration with the rest of the (digital) system.

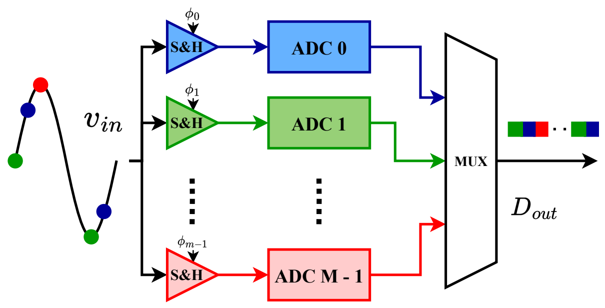

Figure 3: Block diagram of an ideal $M$-channel TI-ADC.

The main issue of such converters is that they are subject to a degradation of the performances due to the presence of mismatches (e.g., gain and offset) between the core sub-converters. In this *thesis* we considered four non-idealities, namely: offset, gain, time-skew, and sub-converter non-linearity. In the next section we will illustrate calibration algorithms for two of them (offset and gain). We decided to model all four non-idealities (instead of just offset and gain) to better characterize the reliability of the on-FPGA model.
Let's consider a generic $M$-channel TI-ADC converter, with a sampling frequency (of the whole interleaved converter) equal to $f_S$, and a sampling time $T_S$.

- The **offset** error of an ADC is defined as the difference between the first ideal code transition and the first actual code transition. In principle, in a single-core ADC this does not represent an issue, but, because of the periodic alternation of the channels in the TI ADC, the different offsets of the cores determine a fixed pattern noise, lowering the SNDR and SFDR of the converter [4]. The error in the output spectrum of the TI-ADC due to the offset mismatch is:

$$E_o(f) = \frac{2\pi}{MT_S} \sum_{n=-\infty}^{\infty} O_n \delta\left(f - \frac{n}{MT_S}\right). \quad (1)$$

Thus, the output spectrum of a TI-ADC with an offset mismatch between the channels shows spurious tones at $nf_S/M$. The amplitude of these tones is given by the term $O_n$, which depends on the offset sequence.

- The **gain** error of an ADC is defined as the difference between the last step's midpoint of the actual ADC and the last step's midpoint of the ideal ADC, once offset error is compensated. The gain error is a specific case of transfer function mismatch in which only the DC gain is taken into account. As before, a multiplicative coefficient in a single-core SAR ADC is not an issue, but, because of the periodic alternation of the channels in the TI-ADC, it modulates the amplitude of the signal, lowering the SNDR and SFDR of the converter [4]. Assuming a sinusoidal input at frequency $f_{in}$, it is possible to express the error in the output spectrum of the TI-ADC due to the gain mismatch as:

$$E_g(f) = \frac{\pi}{MT_S} \sum_{n=-\infty}^{\infty} G_n \delta\left(f \pm f_{in} - \frac{n}{MT_S}\right). \quad (2)$$

Thus, the output spectrum of a TI-ADC affected by the gain mismatch presents a series of spurious tones at $nf_S/M \pm f_{in}$. The amplitude of the tones is given by the term $G_n$, which depends on the gain sequence.

- The **time skew** error is caused by the non-ideal shift between the different phases of the sampling clock. In an ideal TI ADC, each channel samples the input signal after a time $T_S$ from the previous one. In a real application, the time interval between consecutive channels is not constant due to the presence of skew between the different phases of the clock [4]. The output spectrum of a TI-ADC affected by time-skew is:

$$\hat{Y}(f) = \frac{\pi}{MT_S} \sum_{n=-\infty}^{\infty} T_n \delta \left( f \pm f_{in} - \frac{n}{MT_S} \right) . \tag{3}$$

Thus, the output spectrum of a TI-ADC affected by time skew shows a series of spurious tones at $nf_S/M \pm f_{in}$, like the error introduced by the gain mismatch. The amplitude of the tones is given by the term $T_n$, which depends on the skew sequence.

- **Nonlinearity** refers to the non-ideal conversion characteristic of the ADC. A nonlinear characteristic introduces spurious harmonics in the output spectrum of the converter. In this *thesis*, the effect of nonlinearity on the TI-ADC has been analyzed using *ad hoc* MATLAB simulations. Figure 4 shows the spectrum of an 8-channel 2-$GS/s$ TI-ADC where the cores are nonlinear.

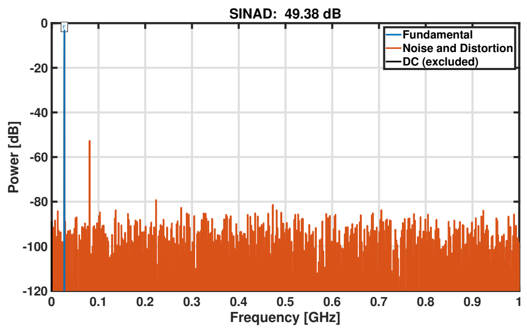

Figure 4: MATLAB simulation results of a 10-bit 2-GS/s 8-channel TI-ADC affected by nonlinearity. The input signal is a sinewave at frequency $f_{in} = 27\ MHz$.

It is possible to notice the presence of a third harmonic at $3f_{in} = 81\ MHz$. The nonlinearity of each core has been modelled according to:

$$S_{nonlinear}[k] = \alpha \cdot tanh \left( \frac{S_{linear}[k]}{\alpha} \right) , \tag{4}$$

where $\alpha$ is a parameter used to control the magnitude of the nonlinearity. For the simulation shown in figure 4, the same $\alpha$ value has been considered for the 8 cores.

## 4. Background Calibration Algorithms

The need for calibration of the mismatches is clear from the analysis presented in section 3. The principal limiting factors on the performances of a TI-ADC are the offset, gain, and time skew. In this *thesis* we focus on the calibration of the first two non-idealities. Both algorithms work by estimating the mismatch of the channel under calibration with respect to a reference one [5].

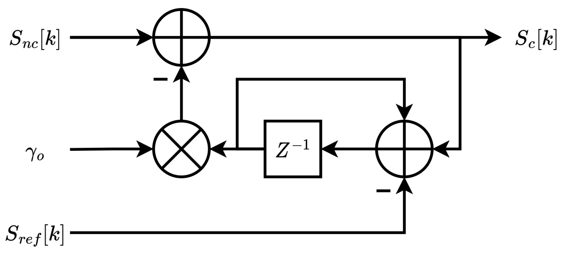

Figure 5: Block diagram of the offset compensation using an IIR filter.

The algorithms make use of an Infinite Impulse Response (IIR) filter (as shown in figure 5 for the offset calibration) to estimate and correct the mismatch. For instance, the working principle of the offset calibration algorithm is described by the following equation:

$$S_c[k] = S_{nc}[k] - \gamma_o \cdot o[k] , \tag{5}$$

where $S_c[k]$ and $S_{nc}[k]$ are the corrected and non-corrected samples of the channel at time instant $k$, respectively, $\gamma_o$ is a coefficient used to control the stability and the convergence speed of the algorithm, and $o[k]$ is the estimation of the offset mismatch at time instant $k$. Its expression is:

$$o[k] = o[k-1] + S_c[k-1] - S_{ref}[k-1] , \tag{6}$$

where $S_{ref}$ is the sample of the reference channel.
The gain calibration algorithm is described by similar equations. In particular equation 5 becomes a multiplication rather than an addition, i.e.,

$$S_c[k] = S_{nc}[k] \cdot \gamma_g \cdot g[k] , \tag{7}$$

and equation 6 is modified to use the absolute values of the signals in order to estimate the gain mismatch:

$$g[k] = g[k-1] - |S_c[k-1]| + |S_{ref}[k-1]| \,, \tag{8}$$

with $\gamma_g$ in place of the aforementioned $\gamma_o$, and $g$ being the estimate of the gain mismatch of the channel.
Figure 6 shows the evolution of the offset calibration coefficients $\gamma_o o_m$ of an 8-channel TI-ADC simulated using MATLAB.

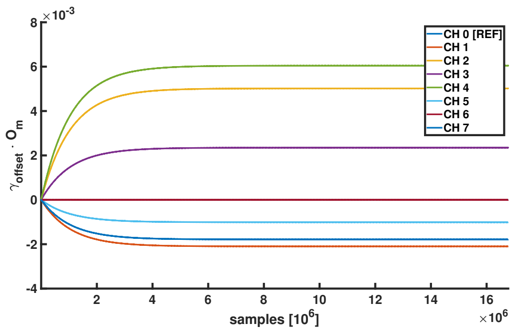

Figure 6: MATLAB simulation of the offset calibration algorithm showing the trend of the $\gamma_o o_m$ coefficients. $\gamma_o = 2^{-16}$.

Both algorithms act entirely in the digital domain, and are thus an ideal target for the proposed verification technique.

## 5. Implementation of the model of the TI-ADC on FPGA

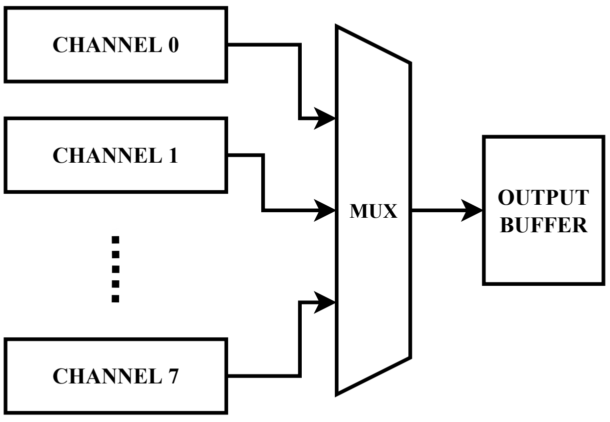

Figure 7: High-level block diagram of the on-FPGA model.

As mentioned in the introduction, the focus of this *thesis* is the verification of the digital calibration algorithms of a TI-ADC on an FPGA. An FPGA is a type of digital IC designed to be configured after manufacturing.
At a high level, the architecture of the on-FPGA model of the target converter (see figure 7) closely resembles the block diagram of the TI-ADC shown in figure 3. In fact, it consists of 8 channels, functionally identical, each composed of a Numerically Controlled Oscillator (NCO), the ADC itself, and the Background Calibration Algorithm (BCA) block. The latter consists of two independent parts correcting the offset and gain, respectively. Figure 8 shows a high-level block diagram of the structure of the channel.

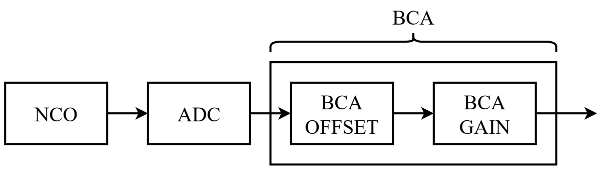

Figure 8: High-level diagram of each channel of the TI-ADC modelled on the FPGA.

- The **NCO** models the Sample & Hold (S&H) circuit. In a real ADC the input signal is analog and time-continuous, properties that are not reproducible in a fully-digital system like an FPGA. The idea is to replace the S&H circuit, which samples the analog, time-continuous signal into an analog, time-discrete one, with an NCO, which directly provides a digital, time-discrete signal. Three different implementations (recursive filter, full Look-Up Table (LUT), and quarter-LUT) have been considered before selecting the full LUT NCO as the most appropriate one. This choice was mainly due to the possibility to model the effect of the time-skew by filling the LUT with a phase-shifted sinewave, which is not possible with the other two implementations.
- The **ADC** models the channel converter and its block diagram is shown in figure 9. This model implements the non-idealities described in section 3. Gain and offset are modelled using a multiplier and an adder, respectively, whereas the nonlinearity of the ADC characteristic is implemented using a LUT. This allows to represent an arbitrarily complex characteristic, or, for instance, to export an ADC characteristic taken

from an EDA tool or even from a measurement of a real device and load it on the LUT to validate the calibration algorithms.

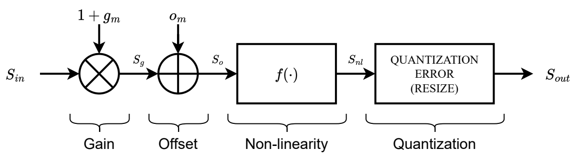

Figure 9: Block diagram of the on-FPGA model of the TI converter.

It is possible to express the resulting output signal of the ADC model as:

$$S_{out} = LUT\left((1+g_m)\cdot S_{in} + o_m\right)\ , \qquad (9)$$

where $1+g_m$ and $o_m$ are the coefficients used to model the gain and offset non-idealities, respectively.

- Finally, the **BCA** block in figure 8 implements the calibration algorithms for gain and offset described in section 4.

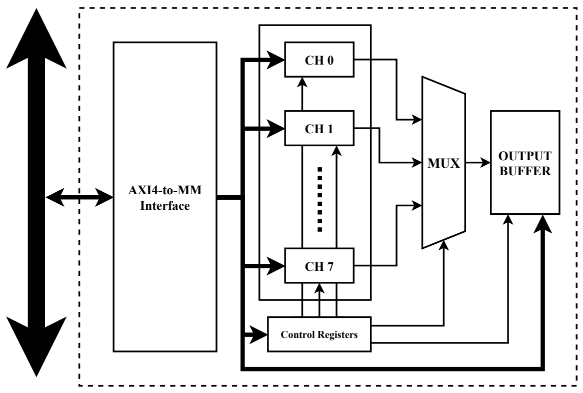

Figure 10: Block diagram of the on-FPGA model of the TI converter.

Aside from the aforementioned blocks, constituting the processing chain of the on-FPGA model of the TI-ADC, we also have some auxiliary blocks (shown in figure 10), namely, the AXI4-to-Memory-Mapped (MM) interface, the reorderer (MUX), the output buffer and the control registers.

- The **AXI4-to-MM interface** is an entity exploited to interface the on-FPGA model of the TI-ADC with the rest of the FPGA. It implements an AXI4 (an industry-standard specification for integrated circuit communication) interface to communicate with the rest of the IPs on the FPGA and translates the write/read operations directed to the on-FPGA model of the TI converter using a simpler (with respect to the complex AXI4 protocol) memory-mapped interface.
- The processed samples are ordered correctly by the **reorderer** (shown as the block named MUX in figures 7 and 10) before being stored in the output buffer. The reorderer is effectively a multiplexer with dedicated control logic.
- The **output buffer** stores the processed samples while waiting for a readout from the PC. This buffer is required due to the limited speed of the available communication interface. The output buffer, and thus the on-FPGA model, can operate in two different modes, namely standard mode, and continuous mode. When working in standard mode, once the memory is full, the on-FPGA model is stopped until the memory is emptied by the PC; When working in continuous mode, the on-FPGA model is never stopped, and the oldest sample is overwritten without stopping the process.
  The continuous mode is particularly important to properly exploit this verification system. In fact, the speed of the communication interface used in this case limits the number of samples per second that can be sent to the PC. Thus, it is either possible to slow down the on-FPGA model to keep up with the interface (i.e., using it in standard mode), or discard most of the samples using the on-FPGA model at its maximum speed (i.e., using it in continuous mode) to verify the real converter in a particular time window.
- Finally, the **Control registers** are used to store the parameters of the on-FPGA model.

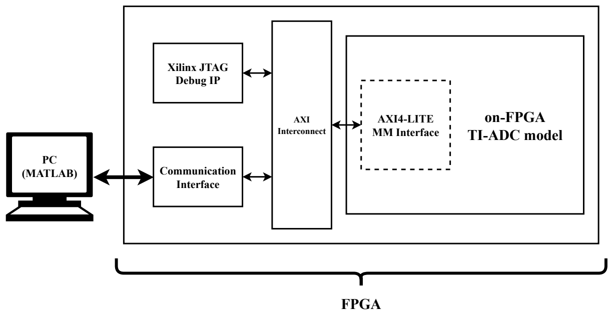

Figure 11: Block diagram of the whole test apparatus.

The implemented model of the TI-ADC has been synthesized on a Xilinx Artix-100T FPGA.

Aside from the on-FPGA model explained in this section, there are also third-party IPs required to communicate and interface with the on-FPGA model. A high-level diagram of the whole design is shown in figure 11.

## 6. Results

Multiple simulations have been performed to compare the results with the ones obtained from MATLAB and VHDL simulations.

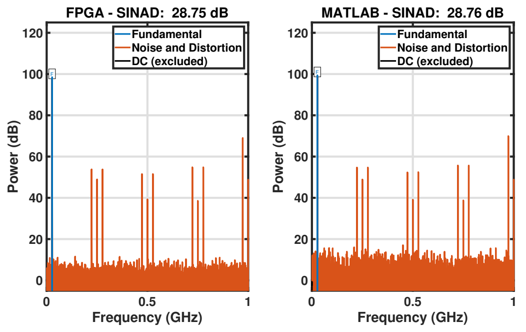

Figure 12: Output spectra of the on-FPGA model (left) and MATLAB model (right). Only offset and gain non-idealities are considered. The input signal is a sinewave at frequency $f_{in} = 27\ MHz$.

Figure 12 shows the spectra of the same simulation scenario for the on-FPGA model (left), and for the MATLAB simulation (right). The accuracy of the on-FPGA model is comparable to the one of the MATLAB model. It is possible to notice how the two spectra in figure 12 are identical for all practical purposes, with a difference in SNDR (SINAD) of only 0.01 $dB$.

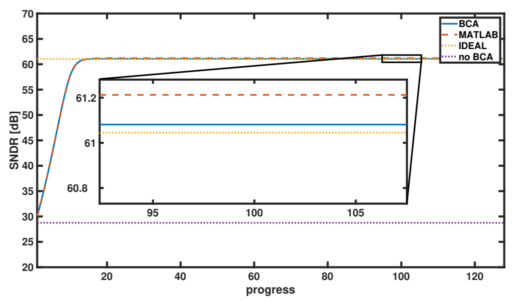

Figure 13: SNDR of the TI-ADC measured every $2^{16}$ samples for a total of $2^{23}$ samples (standard mode). Only offset and gain non-idealities are considered.

This is further confirmed by comparing the evolution of the SNDR of on-FPGA model and the MATLAB one when the BCAs are enabled (as shown in figure 13). We can notice how the two curves showing the progress of the SNDR for the on-FPGA model (in blue) and the MATLAB one (in orange, dashed) are similar, with a maximum mismatch of about 0.1 $dB$. Other simulations have been run considering the different non-idealities analyzed in section 3, both separately and together, showing similar results. Once the reliability of the on-FPGA has been verified, we have performed multiple simulations to characterize the calibration algorithms. Figure 14 shows the result of a simulation performed using the on-FPGA model of the 8-channel TI-ADC affected by offset and gain mismatches, before (left) and after (right) the calibration algorithms convergence.

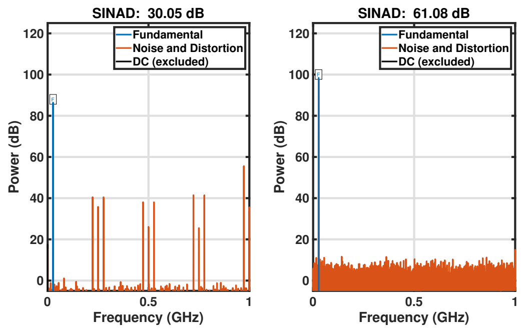

Figure 14: FFT of $2^{16}$ samples of the on-FPGA model, considering the first $2^{16}$ samples (left) and the $128^{th}$ $2^{16}$ samples (right). The input is a sinewave at frequency $f_{in} = 27\ MHz$. Only offset and gain non-idealities are considered.

To compare the performances of the different simulation methods we use a Figure of Merit ($FoM_n$) defined as:

$$FoM_n = \frac{T_{method}}{2\ GS/s}, \tag{10}$$

where $T_{method}$ represents the throughput, meant as the number of samples processed per second of the considered simulation method, and 2-$GS/s$ is the throughput of the TI-ADC we have used as reference.
Table 1 contains a summary of the performances shown by the on-FPGA model compared to simulations run on a PC (MATLAB and behavioral VHDL using ModelSim), and with the real TI-ADC this work aims to model. SM and CM refer to the on-FPGA model used in standard mode and continuous mode, respectively.

| Method | sim. time $[t_{sim}]$ | $FoM_n$ |
|---|---|---|
| FPGA (SM) | 169.1 $s$ | 0.000025 |
| FPGA (CM) | 0.084 $s$ | 0.05 |
| MATLAB | 1.004 $s$ | 0.0042 |
| HDL sim. | 99.4 $s$ | 0.000042 |
| Real ADC | 0.0042 $s$ | 1 |

Table 1: Summary of the simulation times and $FoM_n$ of the verification methods for $2^{23}$ samples, compared to the real ADC.

When used in continuous mode, the on-FPGA model shows a simulation time improvement of a factor 1185 with respect to the VHDL simulations, and of factor 12 with respect to the MATLAB model.
The other advantage of the proposed method is that it allows the verification of the design on real hardware, modelling intrinsically its behavior, like, for instance, saturation or drift of the accumulators or integration of errors caused by the finite precision of digital implementation.
The two advantages can be combined to allow the verification of long-term behaviors of the TI-converter, something that is not possible with VHDL or MATLAB simulations. In fact, VHDL simulations are too slow to detect these errors in a reasonable amount of time, whereas MATLAB is not suited to accurately and realistically describe a digital system of reasonable complexity. Furthermore, in the case of FPGA emulation, the speed of the simulation is independent of the size of the design. Thus, since the number of the resources of state-of-the-art FPGAs closely follow the growth rate of digital electronics (i.e., Moore's law), this verification methodology is able to keep up with the growing complexity of the designs.

## 7. Conclusions

This *thesis* proposes a verification method for the digital calibration algorithms of a TI converter using an FPGA. The main advantage of this technique is the possibility to verify long-term behaviors of the calibration algorithms, which cannot be investigated satisfactorily with standard VHDL or MATLAB simulations.
The proposed methodology has demonstrated a simulation time improvement of a factor 1185 with respect to the VHDL simulations. This improvement factor is limited by the throughput of the available communication interface, which constrains the number of samples per second that can be streamed to the PC. A greater improvement of about 10000 in the simulation time can be reached with a parallel architecture for the TI-ADC model, using all the channels at once. This, however, requires a high-speed communication interface (like for instance, PCIe) to properly exploit the technique.
The proposed method can, in principle, be extended to the verification of the digital sections of any mixed-signal circuit.

## References

[1] Foster. 2020 wilson research group functional verification study, 2020. last accessed on 2022-11-17.

[2] Harry D. Foster. Why the design productivity gap never happened. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 581–584, 2013. doi: 10.1109/ICCAD.2013.6691175.

[3] W.C. Black and D.A. Hodges. Time interleaved converter arrays. *IEEE Journal of Solid-State Circuits*, 15(6):1022–1029, 1980. doi: 10.1109/JSSC.1980.1051512.

[4] Behzad Razavi. Design considerations for interleaved adcs. *IEEE Journal of Solid-State Circuits*, 48(8):1806–1817, 2013. doi: 10.1109/JSSC.2013.2258814.

[5] Asgar Abbaszadeh and Khosrov Dabbagh-Sadeghipour. An efficient postprocessor architecture for channel mismatch correction of time interleaved adcs. In *2010 18th Iranian Conference on Electrical Engineering*, pages 382–385, 2010. doi: 10.1109/IRANIANCEE.2010.5507040.