POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Design of a high-speed asynchronous digital logic for 11-bit 375-Msps SAR A-to-D converter

Laurea Magistrale in Electronics Engineering - Ingegneria Elettronica

Author: Ignacio Pedrero de la Puente

Advisor: Prof. Andrea Bonfanti

Co-advisor: Luca Ricci

Academic year: 2021-2022

1. Introduction

Analog to digital conversion is a critical part of modern electronics. Many digital circuits need fast A-to-D converters to operate. Most of the time, these ADCs are in energy-limited systems where power and area need to be minimized [1]. SAR ADCs are a good solution to operate at relatively high conversion rates (50ksps-500Msps) while keeping an acceptable resolution (8-12 bit) and a low power consumption [1]. Moreover, SAR converters can achieve higher sampling rates by forming time-interleaved architectures, which are made of several converters running in parallel [2]. These architectures keep the same resolution of the single channel SAR, whereas the conversion frequency and power consumption grow proportionally to the number of channels.

A SAR ADC is divided in three blocks: comparator, digital logic and capacitive DAC. The comparator evaluates each bit by comparing a differential input voltage, while the DAC updates this input voltage according to the last comparator decision, so the next bit can be evaluated. The digital logic controls this behavior: resets the comparator after each decision and changes the corresponding DAC switches to update the input voltage, making sure that the comparator is reset until the DAC is settled. It also stores the evaluated bits and displays them at the output when the conversion is finished. The digital logic has an overhead that affects the maximum conversion rate, so many different architectures have been designed to reduce this delay.

In this work, a new digital logic is designed for an existing 11-bit 250-Msps asynchronous SAR ADC, part of a time-interleaved architecture with 8 channels. The proposed digital logic improves the behavior of the converter in terms of conversion rate with the aim of achieving a target operating frequency of 312.5 MHz. However, by modifying the two main loops of the circuit, the implemented logic allows the converter to work at 375 MHz. Also, a control for the comparator input referred noise is implemented, allowing to choose between a low evaluation time for the first (redundant) bits and a more precise comparison for the last (non-redundant) ones.

This work is structured in three main sections. First, the SAR ADC for which this logic has been designed is presented, highlighting its main drawbacks and limitations. Then, the modifications implemented in the logic are detailed,

explaining how they are improving the circuit's performance, and the simulations results are shown. Finally, the results are commented and compared to the project requirements.

2. Existing SAR digital logic

The digital logic is responsible of the control of the ADC operation, resetting the comparator and changing the DAC switches at the right moment along the conversion cycle. It also stores and displays the digital outcome of the conversion. The logic controls the DAC by implementing a binary search algorithm where, after the first bit of the conversion is directly obtained comparing the sampled value on the two halves of a differential DAC, the next steps are followed:

1. The evaluated bit is stored.
2. The comparator is reset.
3. The DAC switch associated to the bit decision is changed, modifying the comparator input voltage.
4. The logic gets ready for the next bit.

The existing logic architecture is shown in Fig. 1. It can be divided in three blocks: DAC control loop, comparator reset loop and auxiliary circuits.

The DAC control loop, made of 13 flip-flops forming a shift-register and 13 latches that control the DAC switches, stores the 13 comparator decisions to generate the digital output value and to implement the monotonic DAC algorithm [1]. After each comparison, the voltage of one half of the DAC has to be reduced by a factor $V_{DD}/2^i$, being i the last evaluated comparison. The half DAC whose voltage is reduced depends on this comparison result: if it is positive, the positive comparator input voltage is shifted, while if it is negative the reduced voltage is the negative one. To implement this, the DAC is controlled by 13 StrongArm dynamic latches (DL 1-13 in Fig. 1). Each of them is triggered by one of the 13 C^2MOS dynamic flip-flops (DFF 1-13 in Fig. 1) that form the shift-register.

The comparator reset loop has to keep the comparator in reset state during the sampling phase and after every comparison, while allowing a new comparison when the DAC voltage is updated. The reset time has to be enough not only to make sure that the comparator is reset, but also that the DAC voltage is settled and ready for the next comparison. The reset state starts thanks to a dynamic latch (DL in Fig. 1) triggered when a bit is evaluated. Then, this latch is reset after the delay of a temporizer. This delay can be digitally trimmed to ensure that the DAC is settled also in presence of process,

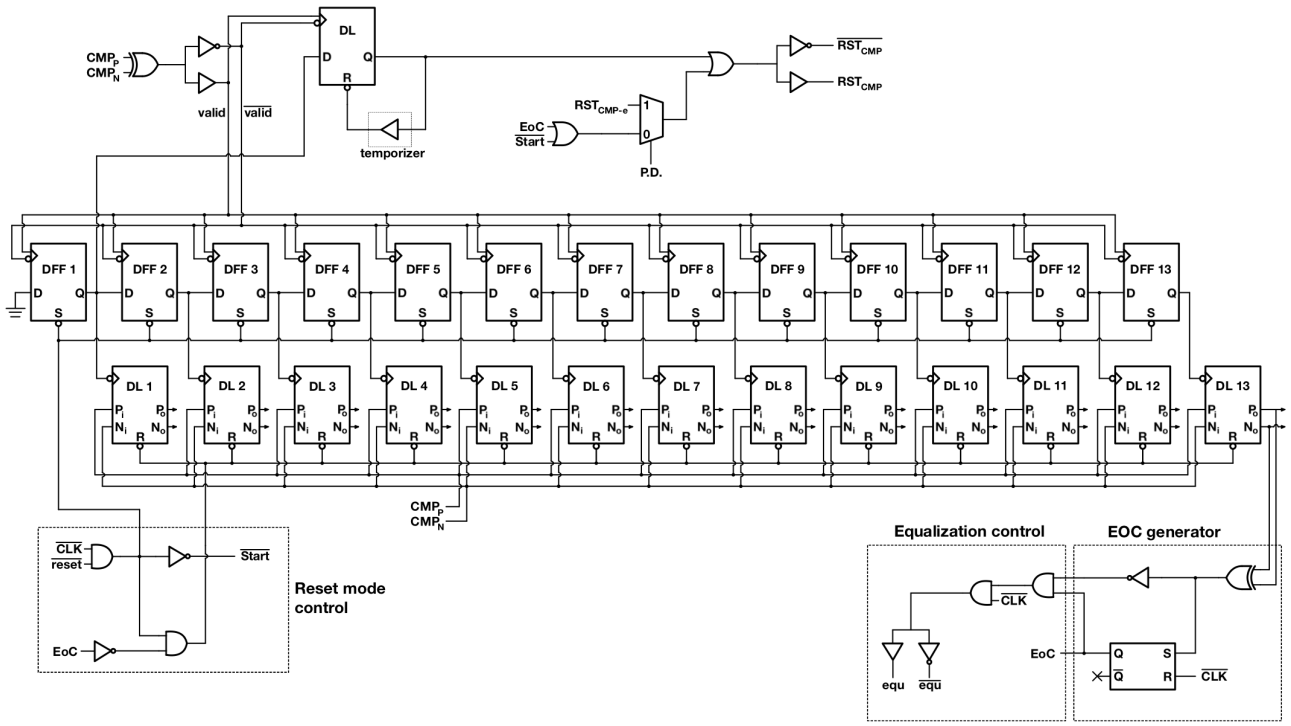

Figure 1: Schematic of the SAR digital logic.

supply voltage and temperature variations. This loop includes also two OR gates and a MUX to reset the comparator during the sampling phase, when the conversion is finished or when it needs to be controlled externally.
Three auxiliary circuits implement additional features needed in the logic. The first is the reset mode control, which resets the comparator and the DLs and sets the DFFs at the beginning of the sampling phase or when the logic is externally reset. It also resets the latches when the comparison is achieved. The second is the end of conversion generator, which generates the signals that reset the comparator, start the reset mode control circuit and display the converted digital value at the output when the last bit is evaluated. The last is an equalization control. When the conversion is finished, the latches are reset to reset the DAC, such that the stored value is the sampled one. However, for the next sampling phase, it is benefitial that the DAC differential voltage before sampling is 0 V. For that reason, an equalization signal is generated, turning on a switch that connects both halves of the DAC.
The SAR ADC exploiting this logic is not able to reach the target conversion rate of 312.5 MHz. The limiting factors of the circuit can be analysed considering the delays in a single comparison step, shown in Table 1. The main loops of the logic work in parallel: both start after the comparator decision and both finish when the temporizer delay is over. Considering a fixed 40 ps comparator decision time, the logic delay represents a 25% of the bit evaluation time in the DAC control loop and a 37% in the comparator reset loop, the DAC settling corresponds to 58% of the bit period and the comparator reset to 46%.

DAC control loop	
Logic delay	DAC settling
57.55 ps	136.15ps
Comparator reset loop	
Logic delay	Comparator reset
86.41 ps	107.29 ps

Table 1: Delay on a loop step in the existing logic.

To achieve the desired conversion speed, the logic delay of both loops and the temporizer (which controls the DAC settling time and the comparator reset time) need to be optimized. The temporizer needs to keep the DAC settling time above a minimum value of 60 ps, thanks to modifications applied in the DAC, whereas the comparator reset time is far from its minimum value and thus it is not a limitation.

3. Proposed SAR digital logic

As the DAC control loop and the comparator reset loop work in parallel, the circuit operation is limited by the slowest path. While the comparator reset time is 40 ps, the original implementation of the DAC takes 120 ps to settle. Thus, it is the latter that limits the circuit operation. The logic overhead for the DAC needs to be reduced in order to achieve higher conversion rates. The main problem of this logic is having to wait for the generation of the valid signal and the delay of a DFF from when a bit is evaluated until it is stored in a DL. In this section, a new architecture is proposed, where the evaluated bit is stored right after the comparator decision.
In the first architecture of the ADC, the time that the comparator is reset is much longer than necessary due to the high DAC settling time, so the logic overhead in the comparator loop is not relevant. However, a new version of the DAC is used in the new converter, and its settling time is reduced to 60 ps. With this new DAC, the comparator reset loop becomes the bottleneck of the digital logic, and thus its delay has to be reduced. Since the main reason for this delay is the number of stages in the path between the comparator outputs and its reset signal, a different architecture is implemented, clearing this path while keeping all the features of the existing logic.
Besides the optimization in terms of speed, another feature is implemented in the logic: a control of the comparator noise. This circuit allows a higher noise in the first, redundant bits, obtaining in exchange a faster comparison delay. Then, the noise is reduced for higher accuracy in the last comparisons, corresponding to non-redundant bits. The specifications of this circuit and the proposed solutions are presented in subsection 3.3.

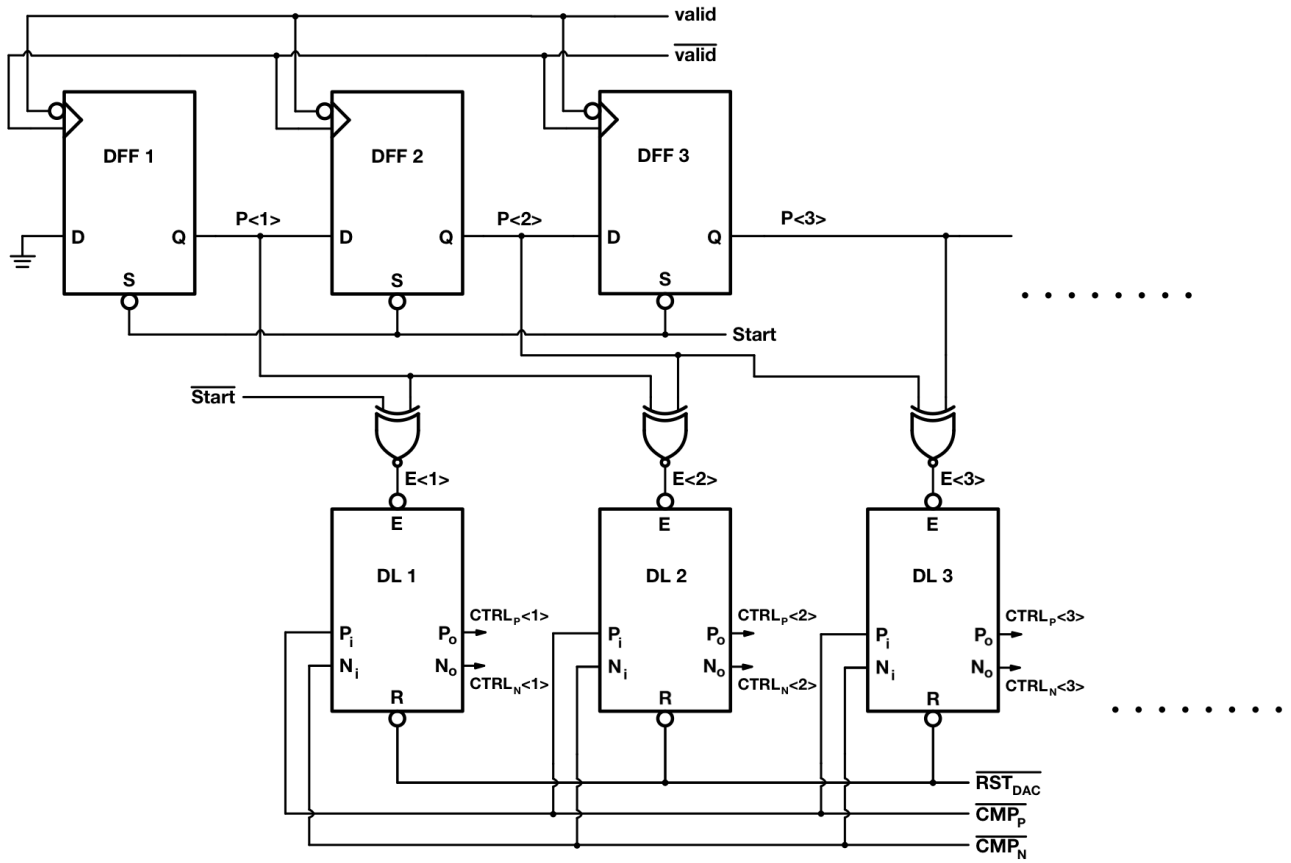

Figure 2: Proposed DAC control loop.

3.1. DAC control loop

The main problem of the existing DAC control loop shown in Fig. 1 is the need to wait for the delay of both the DFF and the DL before starting the DAC settling. Figure 2 shows the proposed solution to this problem, an improved version of the logic presented in [3]. It has a similar structure, with a shift-register of 13 DFFs and 13 latches that store each decision, but now the output of each DFF works as the enable of the corresponding latch instead of being its clock. This way, the DLs are enabled while the comparator is still reset and, when a decision is taken, they are triggered by the comparator outputs, skipping the delay of the flip-flops.

Apart from the fact that the DLs are now enabled, there are some remarkable differences between this logic and the proposed one. First, the DFFs in the shift register are now low-edge triggered, to avoid a situation where a latch is enabled before the comparison corresponding to the previous bit is reset. Second, two flip-flop outputs $P<i>$ are connected to each DL: the first to pull up the enable and the second to pull it down, so just one latch can be enabled at a time. Third, 13 XNOR gates need to be added to the circuit to generate the enables signals. Fourth, the valid signal is now generated with a NAND gate instead of an XOR. This way, a metastable event due to a differential input voltage close to 0 V can also trigger the logic, allowing the system to operate even in the presence of a metastable event (even though the value stored in the latch might be incorrect). Fifth, the outputs of the comparator are now inverted before connecting them to the latch, to adapt the circuit to the used of DLs with enable.

The latch, shown in Fig. 3, is designed with a StrongArm topology and is a modified version of the topology presented in [4]. When the enable is active (i.e., low), M11-M12 are turned on and the comparator outputs are connected to the input transistors M1-M2. Instead, when the enable is high, M13-M14 are turned on, connecting the gate of M1-M2 to ground and turning them off. M5-M6 and M7-M8 implement the bistable element that stores the bit, until it is reset by M3-M4.

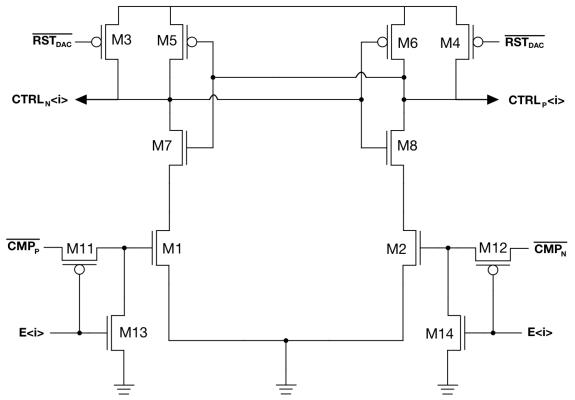

Figure 3: Structure of a dynamic latch with enable.

3.2. Comparator reset loop

The improvements made in the DAC and the reduction of the DAC control loop overhead result in the comparator reset loop being the main limitation of the logic speed. Thus, this paths overhead becomes relevant and need to be reduced. The main problem is that in the original logic, the reset of the comparator occurs after the delay of five stages: an XOR gate, two inverters/buffers, a dynamic latch and an OR gate. A different architecture with a lower logic delay is proposed in [5]. The adapted version is shown in Fig. 4. The path to reset the comparator is now made of just a NAND gate, which allows to reset the comparator also in the case of metastability, a NMOS transistor and a buffer. The temporizer, as in the existing logic, is in the feedback path that stops the reset of the comparator. This path has a similar number of stages to the existing logic, but the delay is reduced due to modifications applied to the temporizer to adapt to the new DAC settling time.

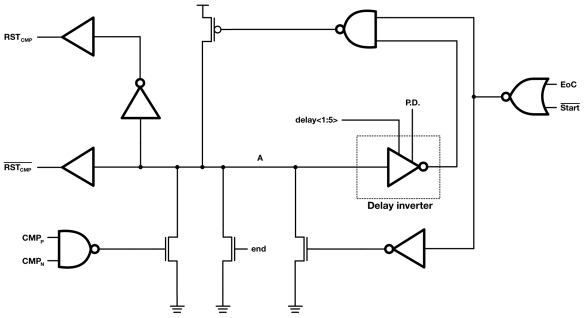

Figure 4: Architecture of the proposed comparator reset loop.

The temporizer needs to reduce its delay to reach the minimum DAC settling time, while keeping the possibility of trimming this delay to compensate variations of process, supply voltage and temperature. In the existing logic it has the behavior of a buffer, while in the proposed one it works as an inverter.

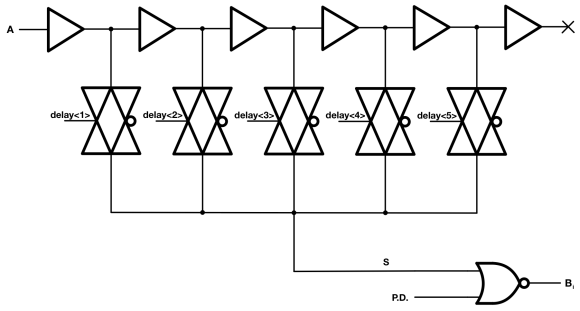

Figure 5: Structure of the temporizer of the proposed logic.

Figure 5 shows the structure of the delay inverter. It exploits buffers and transmission gates to get five different delays to choose from. The NOR gate allows to implement a temporal power down: when *P.D.* is 1, the comparator is kept in reset state; instead, if it is 0, the voltage on the S node is inverted, giving the circuit its inverting characteristic.

3.3. Comparator noise control

The comparator has two stages: a preamplifier and a StrongArm latch. Between this two stages there is a controllable capacitance made of four capacitors connected to ground through a switch which set the speed and the noise behavior of the circuit. At the beginning of the conversion these capacitors should be disconnected to increase the speed of the comparison, since the first bits are redundant and thus noise is not that relevant, while they are connected in the last non-redundant bits evaluation, where the DAC settling time is lower but precision is needed.
The structure of the circuit is shown in Fig. 6. The circuit is made of a 13x1 MUX, a DFF and four 2x1 MUX. Just one *sel<i>* is high, the one corresponding to the bit after which the comparator noise has to be modified. Before this bit is evaluated, *step<i-1>* is pulled up and thus the DFF input is high. When the selected bit is evaluated, this 1 propagates to the 2x1 MUX, so the selected control code (i.e., the number of capacitors that have to be connected) changes from

the initial value to the final one. Both control codes are selected from the output, to adapt to each specific situation. The DFF is reset during the sampling phase so, at the beginning of the conversion, the control code is always the initial one.

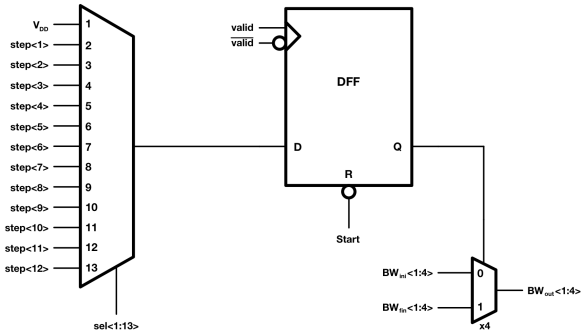

Figure 6: Architecture of the comparator noise control.

3.4. Simulation results

After all the modifications, the behavior of the circuit is analysed and compared to the existing logic. To have a fair comparison, both circuits are simulated under the same conditions: constant differential input voltage of 50 mV, sampling frequency of 250 MHz and a temperature of 100 ºC. The delay of a single comparison step is shown in Table 2. As it can be seen comparing to the results in Table 1, the logic delay has been reduced by a factor of 2 in both loops. In addition, the modifications applied in the DAC allow to reduce its settling time, which is achieved with the new temporizer. Thanks to these improvements, the simulations show that the converter is able to work at a maximum frequency of 375 MHz at schematic level, more than enough to reach the target 312.5 MHz when the circuit is fabricated.

DAC control loop	
Logic delay	DAC settling
29.52 ps	72.63 ps
Comparator reset loop	
Logic delay	Comparator reset
34.81 ps	67.34 ps

Table 2: Delay on a loop step in the proposed logic.

In terms of power consumption, the proposed logic suffers a 30% increase with respect to the existing one, due to both the modified loops and the addition of the comparator noise control. However, the power consumption at the maximum frequency is lower than 2 mW, which is still an acceptable value for the overall converter.

4. Conclusions

In this thesis, a new digital logic for an existing SAR A-to-D converter is presented. Keeping a similar structure, it is able to reduce the delay between the comparator decision and the DAC activation thanks to a smart implementation of the shift-register and the dynamic latches. The improvements made in the DAC allow to reduce the delay of the temporizer and achieve even faster conversion rates. The delay between the comparator decision and its reset has also been reduced adapting a state of the art architecture to the necessities of this logic. Finally, a new feature has been implemented to optimize the comparator operation, and other minor modifications have been made to avoid problems related to metastability.
The simulations show a robust circuit that achieves a big improvement in terms of maximum conversion frequency, outperforming the target value, with a mild increase in terms of power consumption. The logic also gives space to further optimization in the DAC and the comparator.

References

[1] P. Harpe, "Successive approximation analog-to-digital converters: Improving power efficiency and conversion speed," *IEEE Solid-State Circuits Magazine*, vol. 8, no. 4, pp. 64–73, 2016.

[2] S. Louwsma, E. Van Tuijl, and B. Nauta, *Time-interleaved Analog-to-digital Converters*. Springer Science & Business Media, 2010.

[3] Y. Cao, Y. Chen, Z. Ni, F. Ye, and J. Ren, "An 11b 80ms/s sar adc with speed-enhanced sar logic and high-linearity cdac," in *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 18–21, IEEE, 2018.

[4] J.-S. Park, J.-M. Jeon, J.-H. Boo, J.-H. Lee,

K.-I. Cho, H.-J. Kim, G.-C. Ahn, and S.-H. Lee, "A 2.2 mw 12-bit 200ms/s 28nm cmos pipelined sar adc with dynamic register-based high-speed sar logic," in *2020 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, pp. 1–2, IEEE, 2020.

[5] Q. Yu, X. Zhou, K. Hu, Z. Huang, H. Chen, X. Si, J. Yang, and Q. Li, "A 9.08 enob 10b 400ms/s subranging sar adc with subsetted cdac and pdas in 40nm cmos," in *ESSCIRC 2021-IEEE 47th European Solid State Circuits Conference (ESSCIRC)*, pp. 391–394, IEEE, 2021.

POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Design of a high-speed asynchronous digital logic for 11-bit 375-Msps SAR A-to-D converter

Tesi di Laurea Magistrale in
Electronics Engineering - Ingegneria Elettronica

Author: **Ignacio Pedrero de la Puente**

Student ID: 963857
Advisor: Prof. Andrea Bonfanti
Co-advisors: Luca Ricci
Academic Year: 2021-22

Acknowledgements

After almost seven years of university, with a bachelor and two masters, there is many people that deserved to be acknowledge. It is fair to start with the two universities I studied in: Universidad Politécnica de Madrid and Politecnico di Milano. Even though I disagree with how some things are implemented, I must admit that I was able to learn a lot and I am to be an engineer. Some professors deserve special praise, as they have the ability to make me want to learn and not just pass their course. I cannot mention all of them, but I will sure remember what they taught me.

I also want to thank the people in the APPLAB for all their help during these months: Luca Ricci, Gabriele Bè, Lorenzo Scaletti, Luca Bertulessi and professor Andrea Bonfanti. Without their constant help and attention, this thesis just simply would not have happened.

Before starting this master, I had a nice group of friends. After these two years, I understood that I have an amazing group of friends. I will never take for granted again how incredible people they are. However, these two years far from them also allowed me to understand who is the kind of people I want to be with. Some of them are staying here, but I'm sure that we will see each other again.

Between the people I met in Milano, I must mention Paula and Isa for the huge impact they had in my life. I really missed you this year and I can't wait to see you again.

Finally, I want to thank my family. They supported me through everything and help me to achieve my goals no matter what. Staying almost two years far from them was really difficult, and it made me realise that I want to be close to them.

Abstract

Modern 5G and the new 6G communication standards are employed for remote patient monitoring, wearable medical devices, robotics, and telemedicine solutions featuring data rate larger of 1-10Gbit/s. In the electronics circuits supporting these standards, analog-to-digital converter is one of the most demanding blocks due to the high sampling rate. In fact, in this framework, ADCs with a conversion rate greater than 1GS/s and a resolution of at least 10 bits are needed.

Successive approximation register (SAR) converters are a good solution to operate at relatively high conversion rates (50ksps-500Msps) while keeping an acceptable resolution (8-12 bit) and a low power consumption. These converters can achieve even higher sampling rates by resorting to time-interleaved architectures, which are made of several ADC cores running in parallel with time-shifted conversion periods. These architectures keep the same resolution of the single channel SAR, whereas the conversion frequency and power consumption grow proportionally to the number of channels.

The maximum operating frequency of a SAR converter depends on the delay of its main blocks, i.e., the comparator decision and reset time, the DAC settling time and the logic delay. In particular, the logic block, after every conversion decision, toggles the DAC switches according to the comparator output and resets the comparator keeping it reset for the time needed by the DAC voltage to settle. The delay with which these two paths are activated is crucial to enhance the converter sampling rate.

In this framework, this work presents a new architecture for the asynchronous digital logic of an existing 11-bit time-interleaved ADC, made of 8 SAR converters, with the goal of achieving a target operating frequency of 2.5 GHz, i.e., 312.5 MHz for the single core SAR ADC. With respect to the original implementation, the two above-mentioned paths have been sped up: The DAC path benefits from a new topology where the comparator decision directly drives the memory elements that control the DAC switches, and the comparator reset is performed with a reduced number of stages.

The SAR converter, employing this new logic circuit, has been designed in a 28-nm CMOS technology with a 0.9-V supply voltage. From schematic simulations, it features a

maximum conversion rate of 375 MHz, beyond the target of 312.5 MHz, ensuring enough margin to achieve the target operating frequency after fabrication. Moreover, the logic circuit has been supplied with two auxiliary blocks that allow to enhance the original version: a metastability detector and a comparator noise controller. The former circuit allows to solve the problem of metastability, i.e., when the comparator inputs are so close that the decision would take a time larger than the conversion period. In this case, after a fixed amount of time, the bit is set regardless of the comparator decision. The comparator noise control allows to change the comparator noise for the last steps of the conversion, trading off speed with precision.

Keywords: Analog-to-digital conversion, successive approximation register, asynchronous logic, high speed.

Abstract in lingua italiana

Gli standard di comunicazione 5G e 6G sono impiegati nel monitoraggio remoto di pazienti, in dispositivi medici indossabili, nella robotica e nella telemedicina con velocità dati superiori a 1-10 Gbit/s. Nei circuiti elettronici che supportano questi standard, il convertitore analogico-digitale è uno dei blocchi più critici a causa dell'elevata frequenza di campionamento. Infatti, per le applicazioni di cui sopra sono necessari ADC con un tasso di conversione maggiore di 1GS/s e una risoluzione di almeno 10 bit.

I convertitori ad approssimazione successivi (SAR) sono una buona soluzione per operare a velocità di conversione relativamente elevate (50ksps-500Msps) con una discreta risoluzione (8-12 bit) e un basso consumo energetico. Questi convertitori possono consentire velocità di campionamento ancora più elevate se impiegati in architetture "time-interleaved" che sono costituite da diversi ADC che funzionano in parallelo con periodi di conversione "shiftati" nel tempo. Queste architetture mantengono la stessa risoluzione dei singoli convertitori SAR, mentre la frequenza di conversione e il consumo di energia crescono proporzionalmente al numero di canali.

La massima frequenza di conversione di un convertitore SAR dipende dal ritardo dei suoi blocchi principali, ovvero dal tempo di decisione e di reset del comparatore, dal tempo di assestamento del DAC e dal ritardo del blocco logico che governa l'algoritmo a ricerca binaria. In particolare, il blocco logico, dopo la valutazione di ciascun bit, commuta gli interruttori del DAC in base all'uscita del comparatore e resetta il comparatore stesso mantenendolo resettato per il tempo che occorre alla tensione del DAC per stabilizzarsi. Il ritardo con cui questi due percorsi vengono attivati è fondamentale per aumentare la frequenza di campionamento del convertitore.

In questo scenario, il lavoro svolto in questa attività di tesi ha riguardato lo sviluppo di una nuova architettura per la logica asincrona di un ADC a 11 bit "time-interleaved" esistente, composto da 8 convertitori SAR, con l'obiettivo di raggiungere una frequenza di campionamento di 2.5 GHz, ovvero 312.5 MHz per ogni ADC SAR. Rispetto all'implementazione originale, i due percorsi sopra menzionati sono stati velocizzati: Il percorso relativo al pilotaggio del DAC beneficia di una nuova topologia in cui la decisione del comparatore

aziona direttamente gli elementi di memoria che controllano gli interruttori del DAC e il reset del comparatore viene eseguito con una logica che include meno porte logiche.

Il convertitore SAR, che utilizza questo nuovo circuito logico, è stato progettato con una tecnologia CMOS a 28 nm e una tensione di alimentazione di 0.9 V. Dalle simulazioni eseguite sullo schematico del circuito, il singolo convertitore SAR raggiunge una velocità di conversione massima di 375 MHz, oltre il target di 312.5 MHz, garantendo un margine sufficiente per raggiungere la frequenza target dopo la fabbricazione.

Inoltre, la logica è stata dotata di due blocchi ausiliari che consentono di migliorare la versione originale: un rilevatore di metastabilità e un controllore del rumore del comparatore. Il primo circuito permette di risolvere il problema della metastabilità, cioè quando gli ingressi del comparatore sono così vicini che la decisione richiederebbe un tempo maggiore del periodo di conversione. In questo caso, dopo un tempo fissato, il bit viene impostato indipendentemente dalla decisione del comparatore. Il secondo circuito, invece, consente di modificare il rumore del comparatore negli gli ultimi step della conversione, barattando velocità con precisione. Infatti, gli ultimi bit, coinvolgendo le più piccole capacità del DAC hanno tempi di settling ridotti ma richiedono un'elevata precisione in quanto non ridondanti.

Parole chiave: Conversione analogico-digitale, convertitori ad approssimazioni successive, logica asincrona.

Contents

Chapter 1

Introduction to SAR A/D conversion

1.1. Analog to digital conversion

Analog to digital conversion is a capital part of modern electronics. Most electronic circuit operate thanks to digital processors and memories, because digital signals are well defined and easy to read also in presence of noise. However, a lot of the signals that have to be processed are analog, as sound or electromagnetic waves. Therefore, this continuous input has to be transformed into a multilevel discreet output, made of n binary signals: The higher the number of bits, the larger the resolution of the converter.

Due to their complexity, ADCs are usually designed as integrated circuits with CMOS technology. There are different parameters that describe the behavior of an ADC, but the most important are the following:

- **Bandwidth**: the bandwidth is the range of frequencies at which the converter can operate. It is limited by the maximum sampling frequency of the converter and, depending on the kind of ADC, goes from the tens of kHz to the tens of GHz.
- **Resolution**: set by the number of bits, it is the number of different digital levels into which an analog value can be converted. It is also related to the kind of converter.
- **Accuracy**: noise and distorsion alter the effective resolution of the converter, degrading its accuracy. It is tipically expressed as SNDR (Signal to Noise/Distorsion Ratio).
- **Power consumption and area**: since in a lot of cases, ADCs are used in energy and space limited systems, power consumption and area are important figures of merit of the converters.

There is a wide variety of ADC topologies, each of them with advantages and disadvan-

tages that make them useful for different applications. Some of the most used architectures are the following:

- **Flash ADC**: the input voltage is simultaneously compared to all the voltage levels between digital values in a bank of comparators. The positive comparison corresponding to the highest reference voltage is encoded to obtain the digital output. Flash ADCs can achieve high conversion rates (f>20 MHz) but with low resolution (n<6 bit).
- **Dual-slope ADC**: the input voltage is integrated in a capacitor for a fixed time. Then, the capacitor is discharged with a constant slope until it reaches a reference voltage. The digital value is obtained counting the number of clock cycles the capacitor takes to be discharged. In this converters, low speed is obtained in exchange for high resolution.
- **Pipelined ADC**: this kind of converters operate in more than one step. First, a coarse conversion is done in a first ADC for the initial bits. The obtained digital voltage is subtracted from the input one using a DAC. The result is then sampled for the second step of the comparison, where the last bits are analysed in another ADC. Simultaneously, another analog voltage is being converted in the first part of the circuit. This way, the speed of the system is increased, and fast converters with high resolution can be obtained. A digital logic is required to combine the results of both conversions and generate the correct digital output.
- **Succesive approximation register ADC (SAR ADC)**: a comparator is used to obtain each bit through a binary search algorithm. SAR ADCs can achieve medium resolution at medium/high operating frequencies.
- **Time-interleaved ADC**: this converter exploits several ADCs in parallel to increase the number of conversions per clock cycle. The obtained conversion rate is the one of the single ADC multiplied by the number of them, keeping the same resolution of one converter. However, power consumption is also increased proportionally to the number of converters.
- **Other topologies**: tracking, single-slope or sigma-delta are some examples of other typical ADC topologies.

In this work, a time-interleaved ADC is presented, where SAR ADCs are used. The goal is to obtain a high operating speed: 2.5 GHz, i.e., 312.5 MHz for the single channel, with a resolution of 11 bits. For that reason, the SAR ADC operation has to be described, analysing its different elements and their behavior.

1.2. SAR ADCs

SAR ADCs have been proven to be an optimal solution to operate at medium-high operating frequencies (50 ksps-500 Msps) while keeping an acceptable resolution (8-12 bits) and low power consumption [1]. These architectures benefit from technology scalling and low supply voltage, thanks to the use of simple circuits that have an optimum behavior under these conditions. For higher sampling rates, SAR converters greatly benefit from the use of a time-interleaved architecture, multiplying the speed while keeping the resolution of a single SAR and an acceptably low power consumption. SAR ADCs are usually the most power efficient solution for accuracies between 40 and 70 dB of SNDR [2].

In this section, the search algorithm that implements a SAR ADC is described, highlighting the difference between synchronous and asynchronous conversion and describing metastability, an important problem that affects SAR ADCs operation.

1.2.1. Working principle

In a single-ended SAR ADC, the digital conversion of the analog input is obtained with a binary search algorithm, with as many steps as the converter bits. The search tree, with an example case, is depicted in Fig. 1.1.

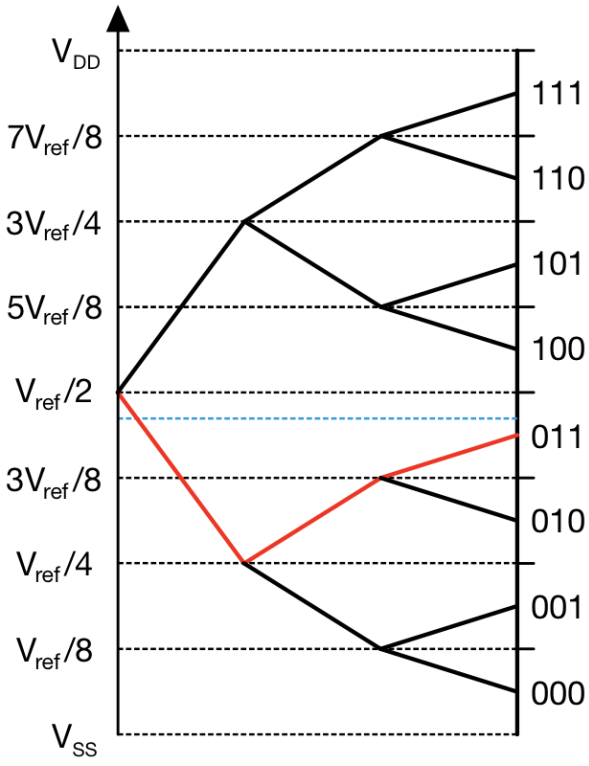

Figure 1.1: Search tree of a single-ended SAR ADC.

The search process goes as follows:

- First, the input analog voltage V_{in} between the positive limit voltage V_{DD} and the negative one V_{SS} that is being converted is compared to a reference voltage $V_{ref}/2$ initially set to $(V_{DD}+V_{SS})/2$, i.e., the mid-range voltage.
- If $V_{in}>V_{ref}/2$, the bit is evaluated as a 1; otherwise, it is evaluated as a 0 (as it happens in the example shown in Fig. 1.1).
- The reference voltage is shifted to the middle of the remaining search range according to the comparison decision, i.e., if the bit is evaluated as 1, $V_{ref}/2+(V_{DD}$ - $V_{SS})/4 = 3V_{ref}/4$, while if it is 0, $V_{ref}/2$ - $(V_{DD}$ - $V_{SS})/4 = V_{ref}/4$.
- The process starts again: the input voltage is compared to the reference one, the corresponding bit is obtained and the reference voltage is shifted by a factor $(V_{DD}$ - $V_{SS})/2^{i+1}$, being i the bit that has just been evaluated.

For differential architectures the process is very similar, but instead of comparing with a reference voltage updated after every bit evaluation, the two input voltages are compared, and one of them is updated by a factor $(V_{DD}$ - $V_{SS})/2^{i}$, depending on the comparison result.

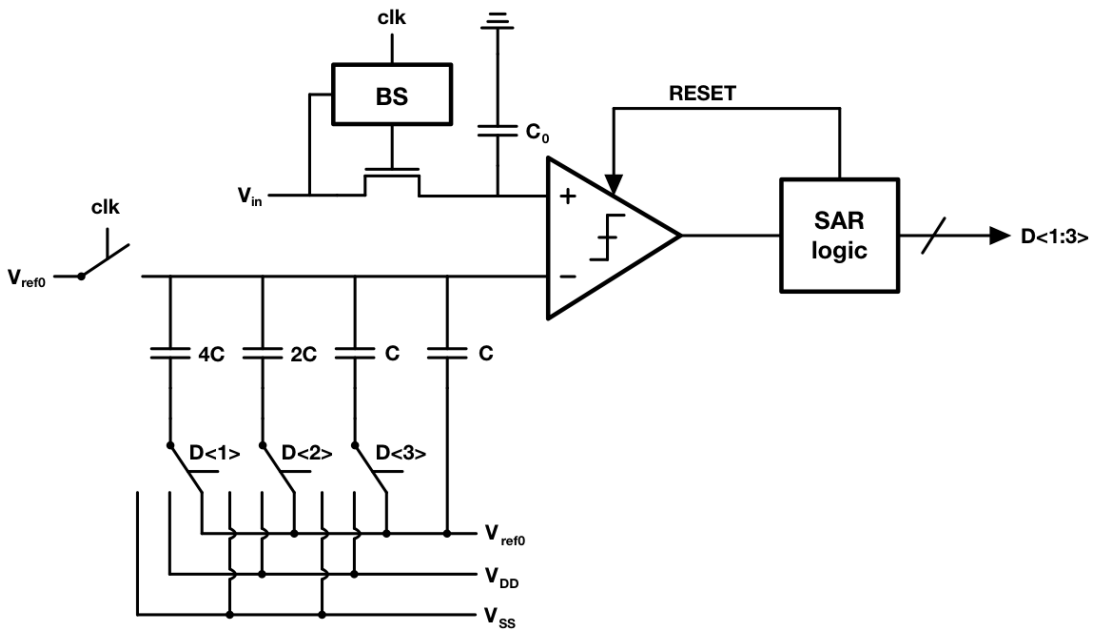

Figure 1.2: Structure of a single-ended SAR ADC.

Figure 1.2 shows the structure of a single-ended SAR ADC. To achieve the described behavior, four different components are needed. First, a sample and hold circuit stores the

input value and keeps it for the whole conversion period. Second, a comparator evaluates each bit comparing the input and reference voltages. Third, a DAC is needed to correctly update the corresponding input voltage between comparisons. Fourth, these operations need to be controlled by a digital logic, which also displays the digital value at the end of the conversion. In this example, the DAC is connected to the reference voltage to apply the described binary search algorithm; however, connecting it to the input voltage allows to use it also to store the sampled value, keeping the negative terminal of the comparator at a constant V_{ref0} voltage.

The sampling and hold circuit is made of an NMOS switch and a bootstrap circuit. With this structure, the gate voltage of the NMOS during the sampling phase is set to $V_{DD}+V_{in}$. Thus, the gate-source voltage is always set to V_{DD}. This allows to avoid problems related to linearity since the settling time, and thus the accuracy of the sampled voltage, would depend on the input signal for a classical NMOS transistor switch.

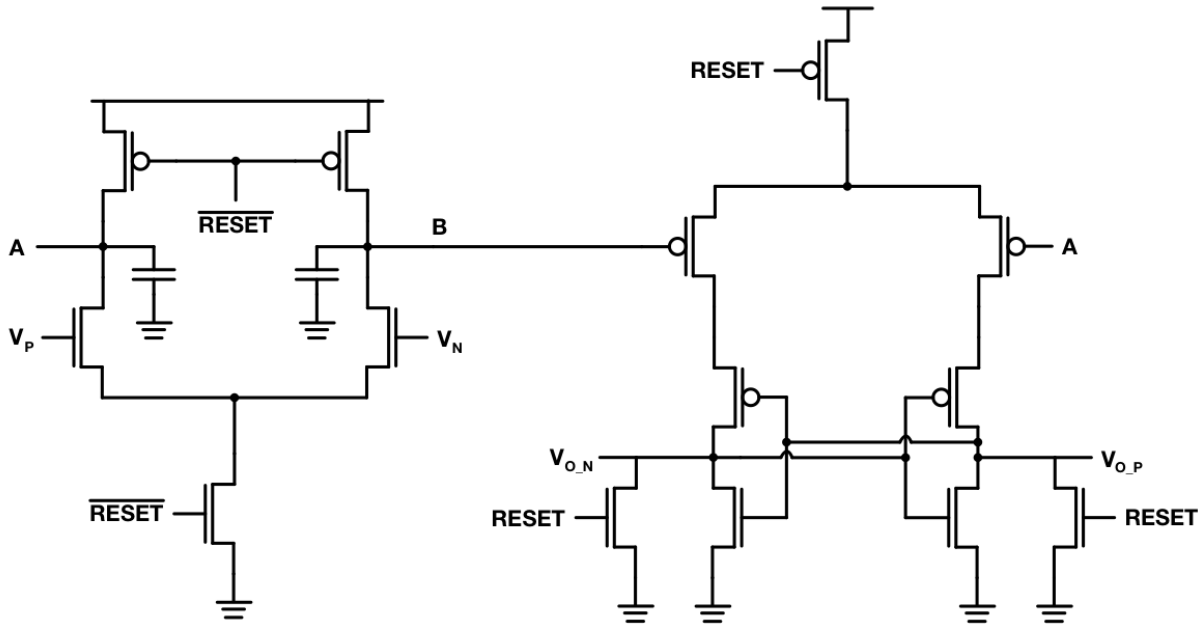

Figure 1.3: Architecture of a classical comparator.

The comparator is the stage that generates a digital output from the analog inputs. A classical dynamic comparator architecture is shown in Fig. 1.3. It includes two stages: a preamplifier and a latch. During the reset phase of the comparator, the capacitors at the output of the preamplifier are charged to V_{DD} and the latch is off. As soon as the reset goes low, these capacitors start discharging. As the input voltages are different, the capacitors discharge at different rates, generating an unbalance at the output of the preamplifier. Then, one of these output voltages turns on a PMOS transistor in the latch.

The latch maximizes the unbalance generated in the preamplifier thanks to the positive feedback.

Both stages need to be reset between consecutive comparisons, with the corresponding delay. The comparator is affected by noise, which can produce incorrect results for low differential input voltages. The input voltage is also related to the speed of the comparator: large input voltages are compared faster, whereas differential voltages close to 0 V need more time to be evaluated.

The DAC updates the value of the reference voltage between comparisons, depending on the result of the last evaluated bit. The most common architectures are based on a differential switched-capacitor network, where the top plate of the capacitors is connected to the inputs of the comparator and the bottom plate can be connected to V_{DD} or ground through a switch. In the most basic structures, the capacitors are binary weighted, i.e., if the capacitor corresponding to the LSB has a capacitance C, the next is $2C$, then $4C$..., and the switching algorithm is monotonic [2], i.e., at the beginning of the conversion, all the capacitors are connected to V_{DD} and then, depending on the comparison result, the switch of the capacitor associated with the evaluated bit is changed in one of the networks, connecting the corresponding capacitor to ground and updating the corresponding voltage, while the other network is kept unchanged. Then, the updated voltages are compared and the process is repeated. Since after every DAC update another bit can be evaluated, the ADC resolution is always one bit higher than the DAC one.

An important problem that occurs in the DAC is capacitor mismatch. The binary scale of the capacitances allows to apply the desired shift to the corresponding input voltage after each comparison. If any of the capacitors size is slightly different to the expected one, the voltage shift after that step is incorrect and can result in a wrong digital output. Mismatch generates a trade-off in the capacitor sizing: small capacitors allow fast operation and low power consumption, but increase the possibility of mismatch.

The digital logic controls the operation of the circuit: starts and stops the reset of the comparator, updates the DAC with the comparator decision and keeps the system on hold until the new DAC voltage is settled. While there is a wide variety of implementations of this logic, they are usually based on a counter and a set of memory elements for the control of the DAC. The counter keeps track of the bit that is currently being evaluated, activating the corresponding memory element that stores the comparator decision and changes the corresponding DAC switch. These memory elements also store the converted digital output.

There are two main kinds of digital logic: synchronous and asynchronous. The syn-

chronous logics are controlled by an external clock signal that allows a fixed time for every comparison when it is high and resets the comparator when it is low. The reset time must be enough for the DAC voltage to settle. The clock signal also triggers the counter. On the other side, asynchronous logic controls the reset of the comparator with a temporizer, so that the comparator is reset from the moment a bit is evaluated and until the DAC is settled. The counter is also triggered after every comparator decision by the same signal that starts the temporizer. This implementation can achieve higher conversion rates, since the comparator needs to operate just for the time the input voltage requires in each specific situation, while in synchronous logics the comparator has to be active for the maximum time for every comparison (usually, the decision time for a 1/2 LSB differential input).

SAR converters can also benefit from redundancy [3], i.e., evaluating some bits with a higher number of comparisons (for example, 7 bits with 9 comparisons). In this case, the capacitors in the DAC are not binary weighted anymore. The main advantage of redundant converters is that the DAC settling time can be reduced, since a possible error due to short settling time is corrected by redundancy.

1.2.2. Metastability

Metastability is one of the most important problems SAR ADCs face to achieve high operation frequencies. It may happen that the differential voltage at the input of the comparator is close to 0 V. As previously mentioned, the decision time of the comparator is directly related to the input voltage. Thus, a small input voltage would take too much time to be evaluated. Since the comparator has a fixed time to make a decision, the bit is not evaluated. Moreover, the DAC voltage is not updated and the next comparison has to be done for the same metastable differential voltage. Thus, no bit can be analysed and the digital result is not generated.

A proposed solution for metastability is the use of metastability detectors: when the comparison time is higher than a specific value, it is considered a metastable event. The evaluated bit is automatically set to a fixed value (either 1 or 0). Then, the DAC is updated with that value and the comparison can be finished. Thus, there is a 50% probability of having an error, and this error would be of just one LSB.

To sum up, SAR converters exploit the use of simple elements that scale well and benefit from advances in process technology. This makes them useful for power efficient operation at medium and high frequencies, specially when they are part of a time-interleaved architecture. To compare different ADC topologies, the Schreier figure of merit [4] can be

used. This FoM is computed as shown in Eq. 1.1.

$$FoM_S = SNDR(dB) + 10\log\left(\frac{f_s/2}{P}\right) \tag{1.1}$$

The goal of Schreier's FoM is to represent the trade-off between the converter accuracy (expressed in terms of SNDR) and its power consumption. Since the power scales with the conversion rate, it is normalized to the Nyquist frequency ($f_s/2$). The SNDR is evaluated at this input frequency in all the ADCs.

Figure 1.4 [1] depicts the Schreier figure of merit with respect to the maximum operating frequency. SAR designs with a bold line represent time-interleaved architectures, while for the other topologies this is represented with a gray shading. It can be observed that SAR architectures can achieve high power efficiency at medium frequency, while inside time-interleaved architectures they achieve the best power efficiencies at high sampling rate.

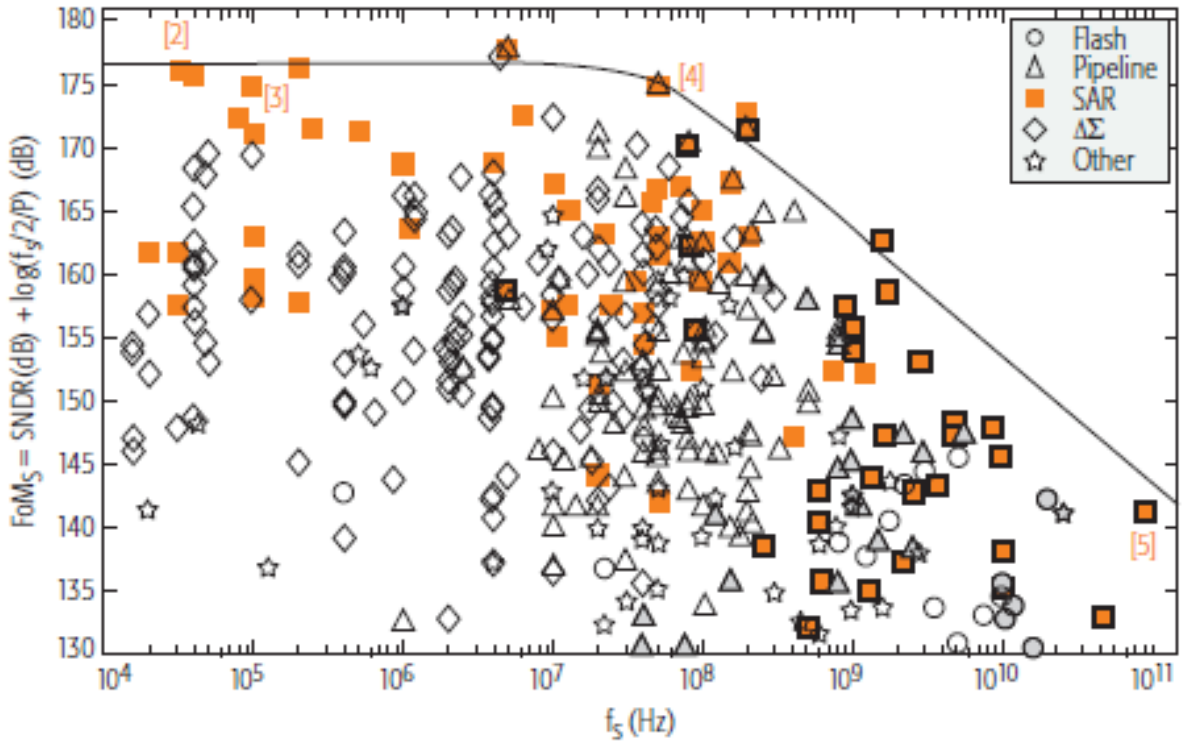

Figure 1.4: Power efficiency of different ADCs vs sampling frequency. SAR converters represented in a bold line or other converters with a gray shading refer to time-interleaved architectures.

1.3. Existing SAR ADC architecture

Figure 1.5 shows the structure of the converter that serves as basis of this thesis. It has a time-interleaved architecture with 8 channels, all with the same SAR structure and working in parallel. The system has a programmable interleaving factor that goes from 2 to 8, i.e., the number of converters that are used in parallel can be controlled. A low interleaving factor can be used to achieve low power operation, whereas higher ones are used to operate at high sampling frequencies. A logic block is required to generate the clock signal of each ADC from the external clock signal and to control the MUX, so the selected input is the last one converted. The clock signals are shown in Fig. 1.6

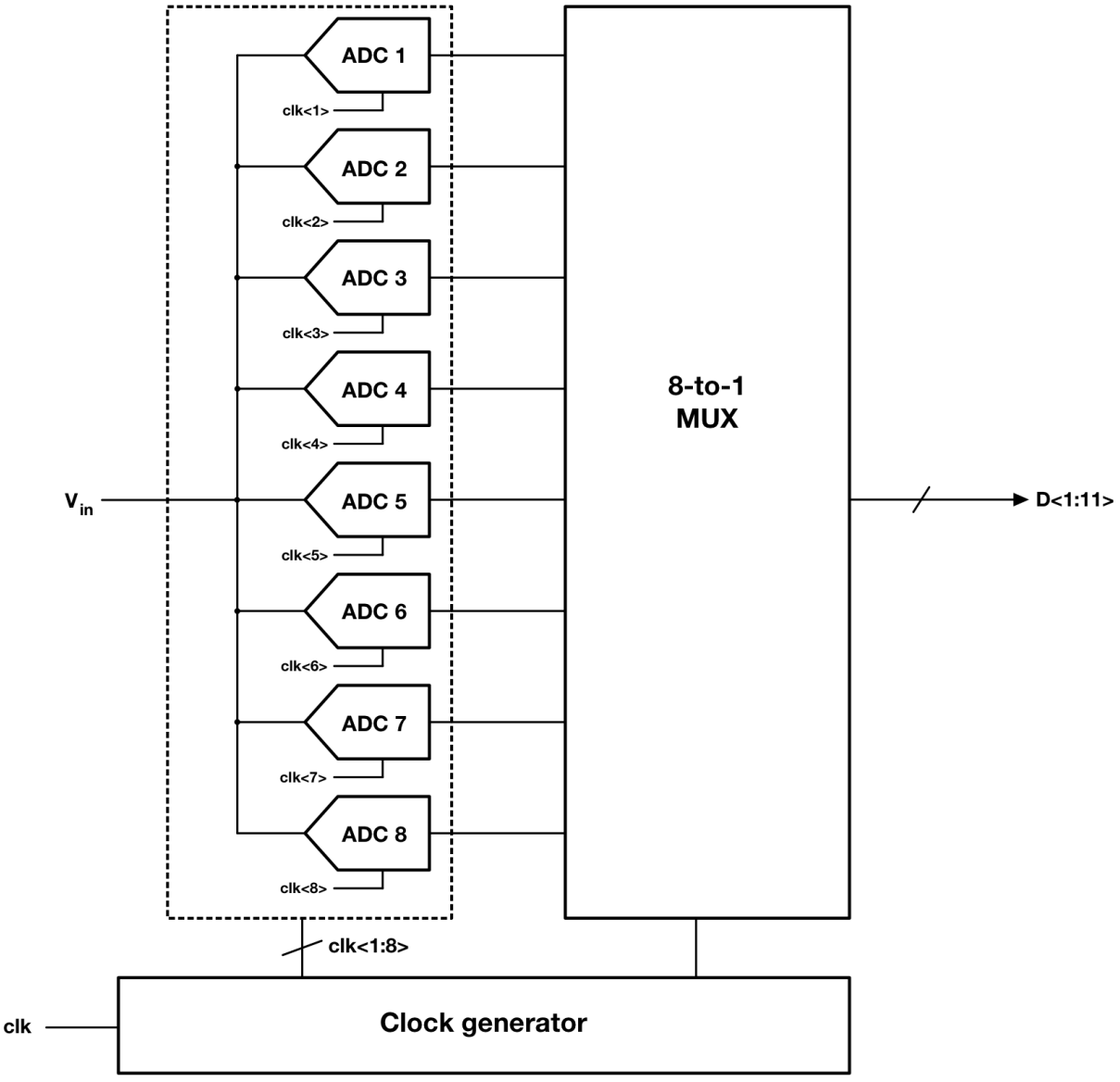

Figure 1.5: Architecture of the time-interleaved ADC.

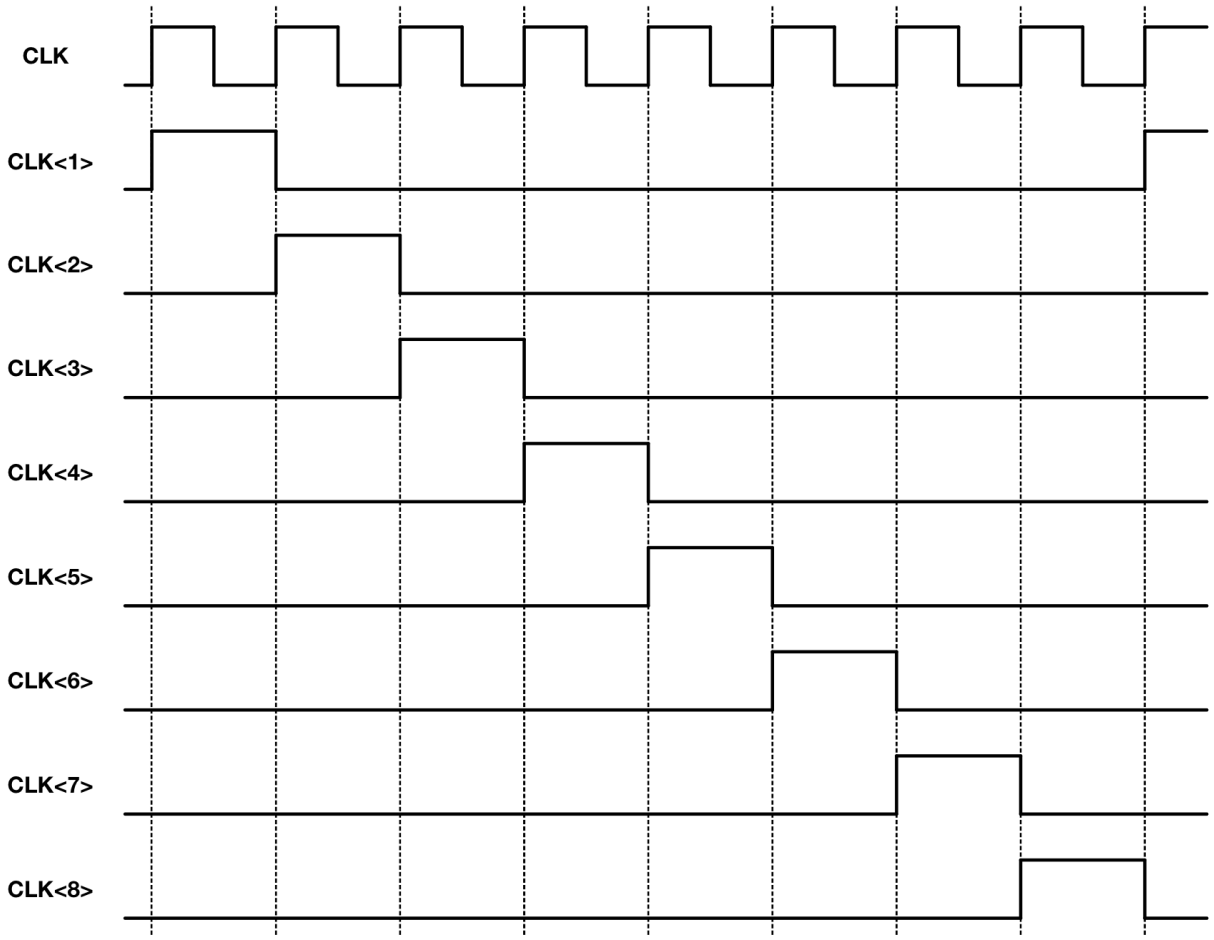

Figure 1.6: Waveform of the clock signals of the eight SAR cores.

Figure 1.7 shows the structure of this SAR converter, which serves as core for the TI structure. First, the sampler is represented by a pair of switches, controlled by the sampling clock *CLK*. Then, the two capacitive arrays (positive and negative) that form the 12-bit DAC are depicted. The switches that update the DAC voltage after each decision are represented by inverters. Each of this inverters is driven by one of the DAC control signals, $CTRL_P$*<1:12>* and $CTRL_N$*<1:12>*. This DAC is not binary scaled in the first steps, but it has an average conversion base of 1.85 to obtain 9 redundant steps that generate the first 7 bits of the digital output [3]. Then, the last 4 DAC capacitors are binary scaled to obtain the remaining 4 bits. The DAC includes the equalizer, represented by a switch and controlled by the *equ* signal. This circuit is activated after the conversion is achieved and the DAC is reset, with the goal of restoring the DAC voltages to the common mode value of $V_{DD}/2$, 450 mV in this architecture. This is done to ensure that the sampling time is always under a limit value.

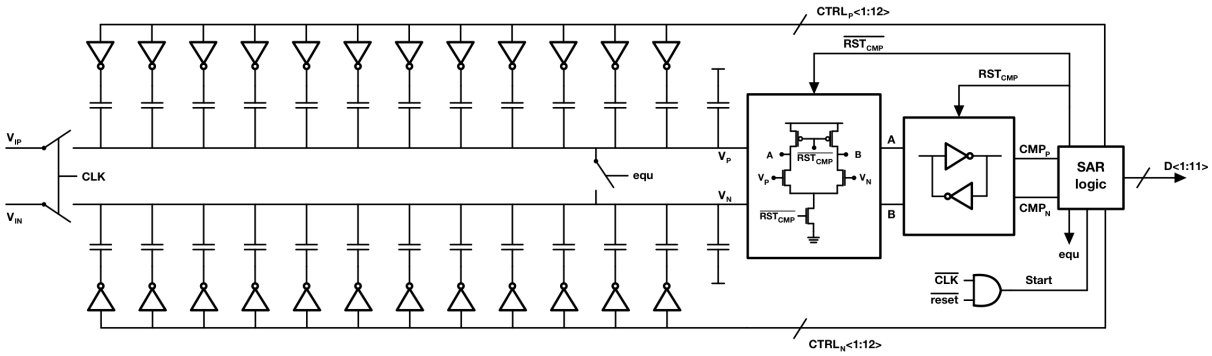

Figure 1.7: Architecture of the SAR ADC.

The circuit implements a quasi-monotonic switching algorithm [5] of the DAC. At the beginning of the sampling phase, the bottom plate of all the capacitors is connected to V_{DD} excepting the one corresponding to the MSB, which are connected to ground. Then, the first bit is evaluated and one of the DAC switches is changed, so the corresponding capacitor bottom plate is connected to V_{DD}. Thus, the voltage of one of the two DACs (i.e., V_P or V_N) is shifted up by a factor $V_{DD}/2$. All the remaining comparisons follow the monotonic algorithm, i.e., after a comparison, the higher voltage is reduced by a factor $V_{DD}/2^{\mathrm{i}}$, being i the last evaluated bit. This quasi-monotonic algorithm allows to reduce the variation of the common mode voltage at the input of the comparator. The highest variations appear in the first steps, where redundancy can compensate for comparison errors, while for the last steps the common mode voltage is close to $V_{DD}/2$.

After the DAC, the figure depicts the two stages comparator. It benefits from the quasi-monotonic switching algorithm, since as the common mode voltage is kept well above ground, a dynamic topology with NMOS input transistors can be used, with advantages in terms of speed and small input capacitance. The first stage is a dynamic preamplifier, which amplifies the input differential voltage. This is done discharging two capacitors through NMOS transistors driven by the input voltages, as shown in Fig. 1.3. The size of these capacitors generates a trade-off in the circuit: big capacitors allow having low thermal noise in the latch, i.e., a more precise comparison, but small capacitors are discharged faster and thus allow higher comparison speed. The second stage is a dynamic latch with PMOS input transistors. Those transistors are off until the voltage of the preamplifier outputs drops below V_{DD} - V_{Tp}. When this happens, an unbalance has appeared in the outputs of the preamplifier A and B, fastly triggering the positive feedback. This allows a faster comparison and reduce the cross-conduction current in the latch, and thus the power consumption.

Finally, at the right end of the figure there is the SAR digital logic that controls the circuit,

and whose optimization is the goal of this work. When a comparator decision is made, the SAR logic activates the reset of both stages of the comparator, through the signals RST_{CMP} for the latch and $\overline{RST_{CMP}}$ for the preamplifier. Simultaneously, it changes one of the DAC switches through $CTRL_P$<1:12> and $CTRL_N$<1:12>. One of the DAC voltages (V_P or V_N) is shifted after the DAC settling time. Until the voltage is completely updated, the comparator is kept in reset mode. The logic includes a temporizer which controls the time duration of the comparator reset. The logic also controls the equalization. When the last bit is evaluated, the DAC is reset and *equ*, the signal that controls the equalization switch, is pulled up.

The SAR logic also need to keep the comparator in reset state and the DAC switches in their initial position either when the converter is sampling (i.e., CLK=1) or when the external reset signal *reset* is activated. For that purpose, there is a signal that takes special relevance: the *Start* signal. This signal is generated with an AND gate whose inputs are the complement of the sampling clock $\overline{CLK}$ and the external reset, and it has great relevance in the operation of the SAR logic, presented in Chapter 2.

For its operation, the digital logic is divided in two loops, one for the reset of the comparator and other for the control of the DAC. Both loops have a delay that limits the operation of the logic in terms of speed. The comparator reset loop has a higher overhead, but the DAC control loop is more relevant due to the high settling time of the DAC.

The described converter is designed in a 28-nm CMOS technology with a 0.9 V supply voltage. Even though the majority of the transistors employed in the design of the logic have a nominal threshold voltage, also low threshold voltage and ultra low threshold voltage transistors have been used in the circuit for specific applications.

1.4. Scope of the work

This thesis has been carrried out in the Department of Electronics, Information, and Bioengineering of the Politecnico di Milano, specifically in the research group of high-speed ADCs for 5G wireless applications inside the "APPLAB". It focuses on the digital logic of a SAR ADC to achieve a target operating frequency of 312.5 MHz, find a solution to the metastability problem and design a circuit that manages a new feature of the comparator, where a controllable capacitance is introduce between the preamplifier and the latch to modify the thermal noise of the comparator.

As already stated, the DAC control loop is the limiting factor of the logic in terms of speed, and thus its optimization is the priority of this work. However, modifications made in the

DAC in a parallel work and the reduction of its settling time have forced to modify also the comparator reset loop, so that this path does not limit the overall conversion time. Moreover, an improvement of the logic that activates both loops has been performed to further enhance the speed of the circuit.

The optimization of the main loops of the circuit needs to take into account the modifications that the logic requires to avoid malfunction in case of a metastable event. Thus, it needs to include a solution for metastability situations that does not affect the overhead of the logic.

Finally, the implementation of the comparator noise control has to find an optimal solution in terms of area and power consumption, so it is profitable to add it to the logic. The goal is to find a circuit that benefits from the use of other signals generated inside the logic and needs a minimum number of elements.

Chapter 2

Existing SAR logic

The SAR logic is responsible of controlling the DAC and the comparator, and it also stores the digital result of the conversion. It controls the DAC implementing a binary search algorithm [2] where, after the first bit of the conversion is directly obtained comparing the sampled value on the two halves of the capacitive array, the next steps are followed:

1. The evaluated bit is stored.
2. The comparator is reset.
3. The DAC switch associated to the bit decision is changed, modifying the comparator input voltage.
4. The logic gets ready for the next bit.

To achieve this behavior, two blocks are always needed: bistable elements to store the evaluated bits and a temporizer, which decides how long the comparator is reset between two operations, and thus how much time the DAC has to settle. The logic has some other functionalities, like starting the conversion after the sampling phase or resetting the comparator and the DAC when a conversion is completed. Figure 2.1 shows the schematic of the logic used as the basis of this work.

As it can be seen in the figure, it is a complex circuit with numerous elements connected with each other. For ease of explanation, the logic has been divided in five different blocks: DAC control loop, comparator reset loop, end of conversion generator, DAC equalization control and reset mode control. The following analysis is focused on each individual block behavior, taking into account also the expected input and output signals. After this, a whole conversion loop is described, highlighting the most relevant signals and their interconnections.

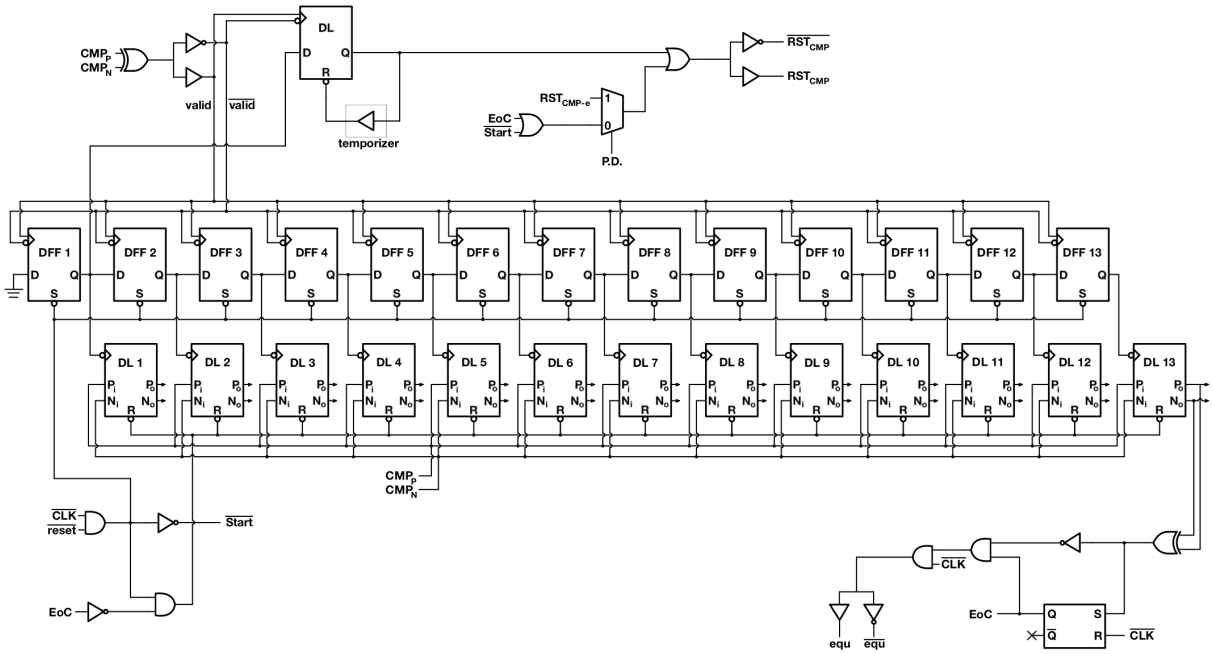

Figure 2.1: Schematic of the SAR digital logic.

It is important to mention that, since the main goal of this logic is achieving high sampling rates, all the different clocked digital stages, as the flip-flops or the latches are designed in dynamic logic.

2.1. DAC control loop

The DAC control loop is one of the principal structures of the digital logic. Its main function is changing the correct switch in the DAC after each comparison, setting the new values of the comparator inputs before the next step. The signals that allow this operation are also the digital outputs of the ADC, i.e., they are the converted digital value. The DAC control loop also gives information about the comparison step that is taking place at every moment of the conversion cycle. This output can be useful in case of malfunction of the system.

To achieve the described behavior, the following inputs are needed:

- **Comparison done** (*valid* and $\overline{valid}$): the signal which triggers every stage of the DAC control loop is the confirmation that the comparison has been achieved. This signal is generated using the outputs of the comparator as inputs of a XOR gate. When the comparison is finished, one of the inputs of the XOR is pulled down, while the other remains high, so the XOR is driven high. This signal is connected then to a buffer and an inverter, to get both an active high and an active low signal when

the comparison is finished.

- **Comparison result** (CMP_P and CMP_N): the input voltage of the DAC needs to change according to the comparison result. If the positive voltage is higher than the negative one, it must be reduced according to the binary algorithm (and vice versa). For that reason, the DAC control loop needs information about the comparator outcome. Specifically, two inputs are needed, corresponding to the positive and negative outputs of the comparator. They are high after the reset, and then one of them is pulled down: CMP_P if the input differential voltage is negative, or CMP_N if it is positive.

- **DAC reset** ($\overline{RST_{DAC}}$): the purpose of this signal, which is active low, is to turn on all the DAC switches, so the comparator inputs go back to the original sampled voltages. Then, the equalization sets them to $V_{DD}/2$. Consequently, the DAC reset is pulled down when the last comparison cycle is achieved, but also when the converter goes into sampling phase or when it is reset from the outside.

- **General set** (*Start*): this input sets the flip-flops. It needs to be activated either when the converter is in the sampling phase or when it is externally reset. In these cases, its value is 0.

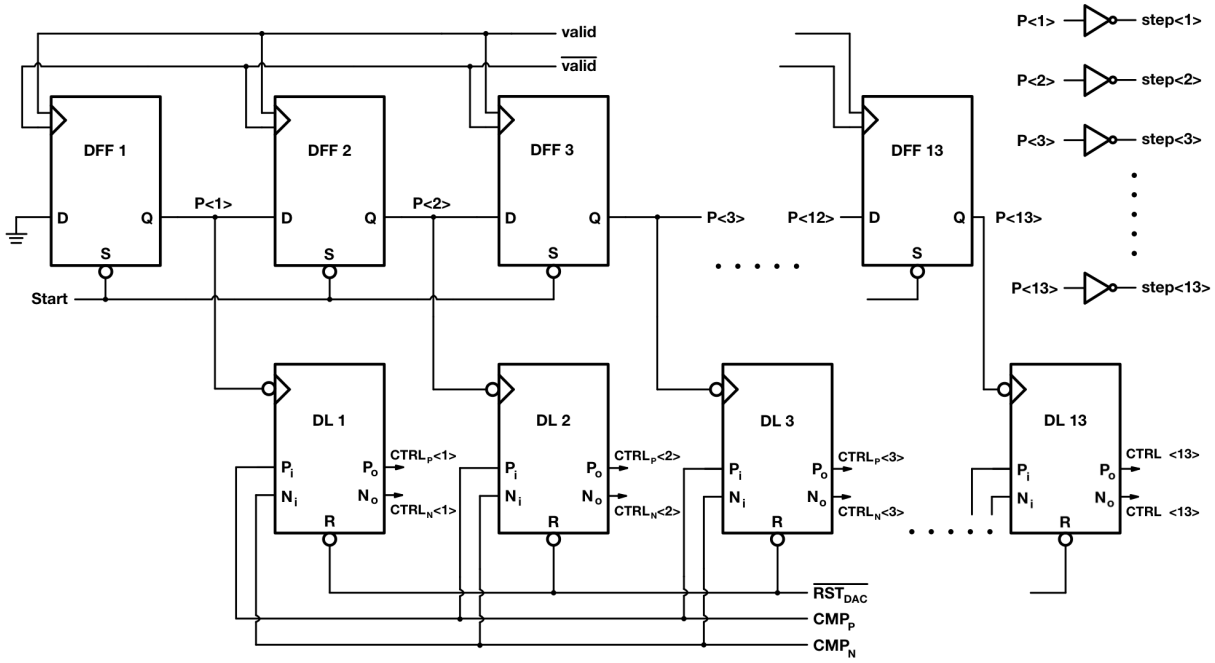

Figure 2.2: Schematic of the DAC control loop.

The structure of the DAC control loop is represented in Fig. 2.2. As it can be seen, it is a shift-register of 13 DFFs [6] with their outputs connected to 13 latches [6]. The DAC

reset is applied to the latches, while the flip-flops are initialised by the *Start* signal. *valid* and $\overline{valid}$ act as the clock for the flip-flops, while the comparison result is stored in the latches.

The goal of this architecture is to activate one of the latches after each comparison, to store its result and update the DAC voltage. For that purpose, the flip-flops implement a shift-register: the output of each flip-flop, called $P<i>$ (it stands for "propagate") is connected to the input of the next one. During the sampling phase, the DFFs are initialised to V_{DD}. The input of the first flip-flop DFF1 in the chain is connected to ground, so its output is pulled down after the first comparison, once *valid* is pulled up. As a consequence, the first latch DL1 is triggered and the input of the second flip-flop is set to 0. This ensures that after every comparison, the corresponding latch, which is negative edge triggered, stores the outputs of the comparator, changing the corresponding DAC switch thanks to its outputs, $CTRL_P<i>$ and $CTRL_N<i>$. The outputs of the flip-flops are also inverted, as it is shown in Fig. 2.2, generating several step signals named $step<i>$, which indicate that the corresponding comparison is completed.

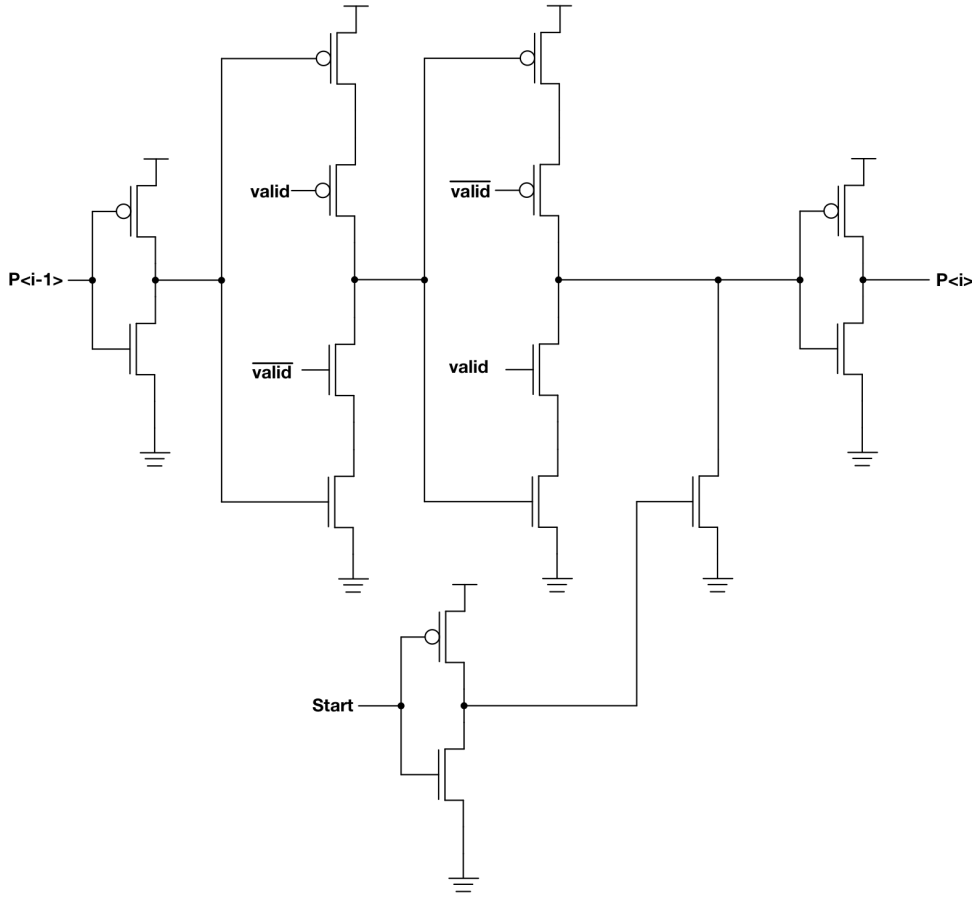

Figure 2.3: Architecture of a C^{2}MOS flip-flop adopted in the shift-register.

Figure 2.3 shows the structure of the flip-flops in the shift-register. As mentioned above, it has been designed in dynamic logic to minimize its propagation delay. It has a classic C^2MOS topology [6]. The active low set *Start* forces $P<i>$ to 1. The input value $P<i-1>$ is inverted and has to wait for the reset to V_{DD} of the comparator outputs in order to propagate through the first tristate inverter, controlled by *valid* and $\overline{valid}$. When a comparison is done, the signal goes through the second tristate inverter and the output inverter and sets the voltage of the output. $P<i>$, high at the beginning, is pulled down only once during the conversion, triggering every latch only once as well.

The latch function is storing the result of each comparison. With that information, the converter needs to switch the related DAC capacitor and generate the digital output at the end of its operation. Due to the first feature, the latches must be differential, with the same output voltages in reset state and with one of them switching according to the result of the comparison.

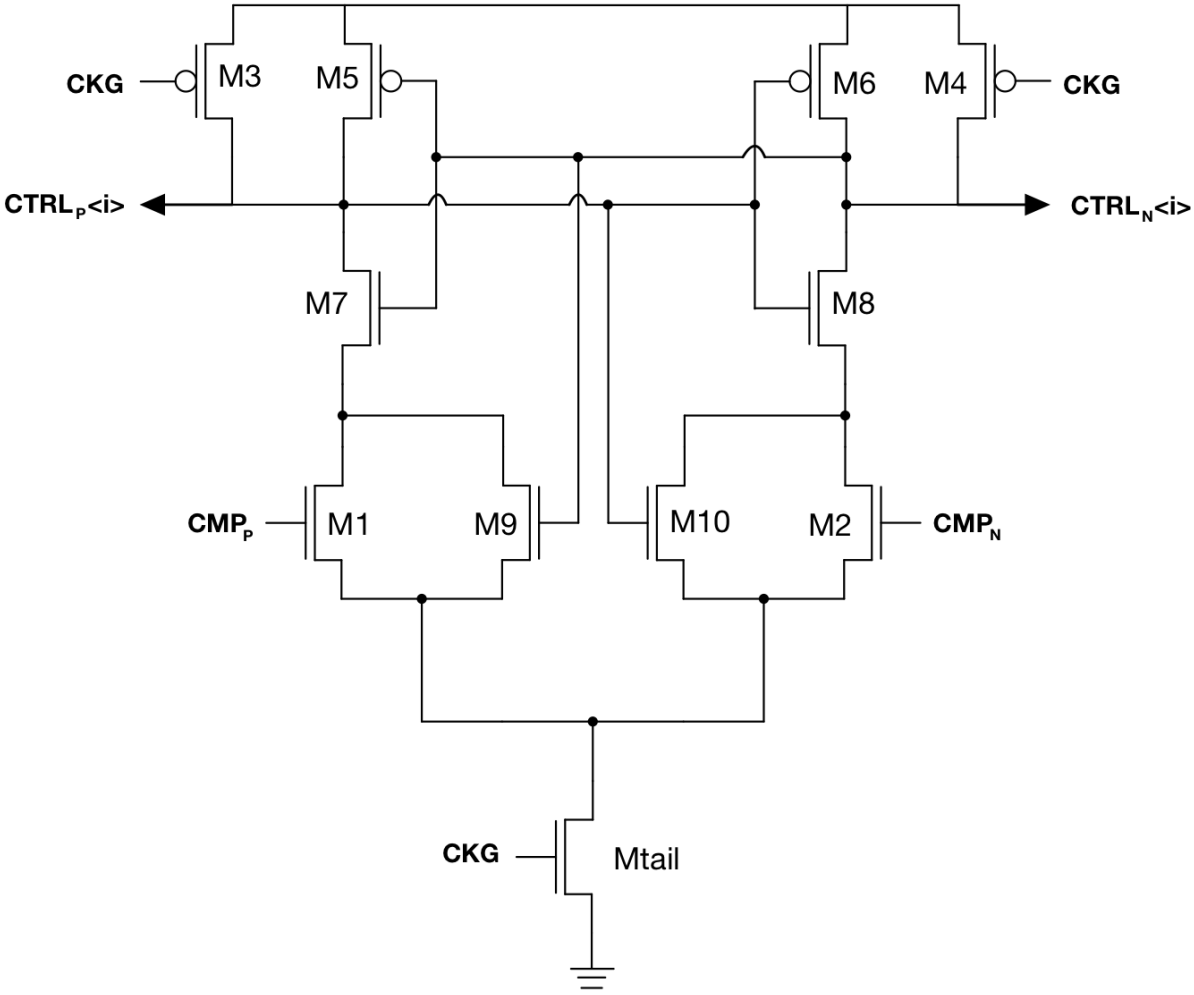

Figure 2.4: Architecture of a StrongARM latch.

To achieve that behavior, the latch is designed with a StrongARM topology [6], as shown in Fig. 2.4. The PMOS transistors M3 and M4 are responsible of keeping the output node at V_{DD} in reset state, while the NMOS Mtail allows the latch to turn on. When it is active, both output nodes start discharging through M7-M8 and M9-M10, but also through M1 or M2, depending on the comparator decision. Due to this unbalance, one of the output nodes discharges faster. If CMP_P=0 (CMP_N=0) the gate voltage of M5 (M6) drops turning it on, pulling up the gate voltage of M6 (M5) keeping it off. It also keeps M8 (M7) and M10 (M9) on, while the complementary ones turn off when their gate voltage is lower than the threshold. At the end, $CTRL_P$ ($CTRL_N$) is connected to V_{DD} through M5 (M6), while $CTRL_N$ ($CTRL_P$) is connected to ground through M8 (M7), M10 (M9) and Mtail. The function of M9-M10 is creating a low impedance path between the discharged output and ground, because M1-M2 are going to be turned off during the next comparison steps.

When a latch is triggered, the value of the comparator outputs in that moment is stored discharging one of the output nodes. Even though the latch is in transparent mode until the end of the conversion, the stored values cannot change even if the comparator result changes. This behavior, which ensures the correct operation of the logic, is due to the discharge of the corresponding output node: since transistors M1 and M2 can only create a path from the output node to ground, under no circumstances this output node can be charged due to a variation in the comparator outputs. The other node, the one that is high, can't be discharged either, since M7 (M8) is turned off, breaking the path between the output node and ground. Therefore, the value stored in the latch remains until the end of conversion, keeping the DAC switches to the desired position.

The reset phase and the latching one are triggered by the same signal, called *CKG*. The goal of this configuration is avoiding to turn on the tail transistor while the DL is in reset state, which would cause a cross-conduction current. Also, having the latch in reset state until it is triggered means that the outputs are always connected to V_{DD} through a low impedance path, avoiding voltage reductions due to leakage currents.

The clock of the latch (*CKG* in Fig. 2.4) needs to take into account both the active low reset input signal $\overline{RST_{DAC}}$ and the propagate one $P<i>$. If the input reset is active or $P<i>$=1, *CKG* must be 0, while it is pulled up when none of the previous conditions are met. This behavior can be obtained with a NOR gate, as shown in Fig. 2.5. $\overline{RST_{DAC}}$ is inverted before being applied to the input, while the $P<i>$ is connected directly.

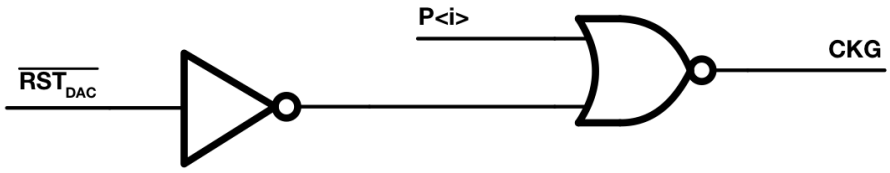

Figure 2.5: Generation of the latch CKG signal.

It may be necessary to turn on a specific switch of the DAC from the outside. That is done by connecting the outputs of the latch to two AND gates. If the other input of the gate, externally controlled, is 1, the AND gate acts as a buffer and therefore the output value is the expected one. Instead, if the other input is 0, the output is also 0, switching from V_{DD} to ground the bottom plate voltage of the corresponding DAC capacitor.

2.2. Comparator reset loop

The comparator reset loop is the other main structure of the logic. It controls whether the comparator is kept in reset state or it is comparing the two inputs during the conversion process. When the ADC is sampling or when the conversion has successfully ended, the comparator must be reset. When the conversion is in progress, it has to be reset after the comparison result is stored in a latch, and kept in that state for the time needed by the DAC to settle. This time can be trimmed to compensate the variations that might occur due to PVT, ensuring the correct behavior of the whole system. In nominal conditions, and therefore in the simulations, an intermediate delay should be used, while the faster and slower ones are used when the conditions are different than the expected.

This complex subsystem is triggered by several different situations. Because of this, it needs various inputs, described below:

- **Comparison done** (*valid* and $\overline{valid}$): the confirmation that the comparison is finished has to reset the comparator. As already explained for the DAC control loop, it is generated using the outputs of the comparator as inputs of an XOR gate.
- **End of conversion** (*EoC*): this signal is pulled up when the last comparison is completed and until the beginning of the next sampling phase.
- **General reset** ($\overline{Start}$): as for the DAC, the comparator needs to be in reset state either in sampling phase or when the external reset is activated. As opposed to the DAC control loop, this signal is active high. As general reset, $\overline{Start}$ is exploited, being *Start* the signal that marks the start of the conversion.

- **First step** (*P<1>*): even though the trigger of the subsystem in normal functioning is the comparison done signal, the DAC needs some extra time to settle the value of the input voltage after the first comparison. This is because the capacitor to be charged is the biggest of the array. As a consequence, this signal is pulled up at the same time the first latch in the DAC control loop is triggered.
- **Reset time control** (*delay<1:5>*): this five-bit input trims the time the comparator is reset between consecutive operations. It gives the opportunity to choose between five different delays. Just one of the five bits can be set to 1, while the others must be kept to 0.
- **Temporal power down and external comparator reset** (*P.D.* and $RST_{CMP\text{-}e}$): the temporal power down *P.D.* disables the comparator reset loop operation, and keeps it in reset state with the help of the external comparator reset $RST_{CMP\text{-}e}$. This last signal is different from the general external reset, which turns off the whole logic whereas this one just controls the comparator.

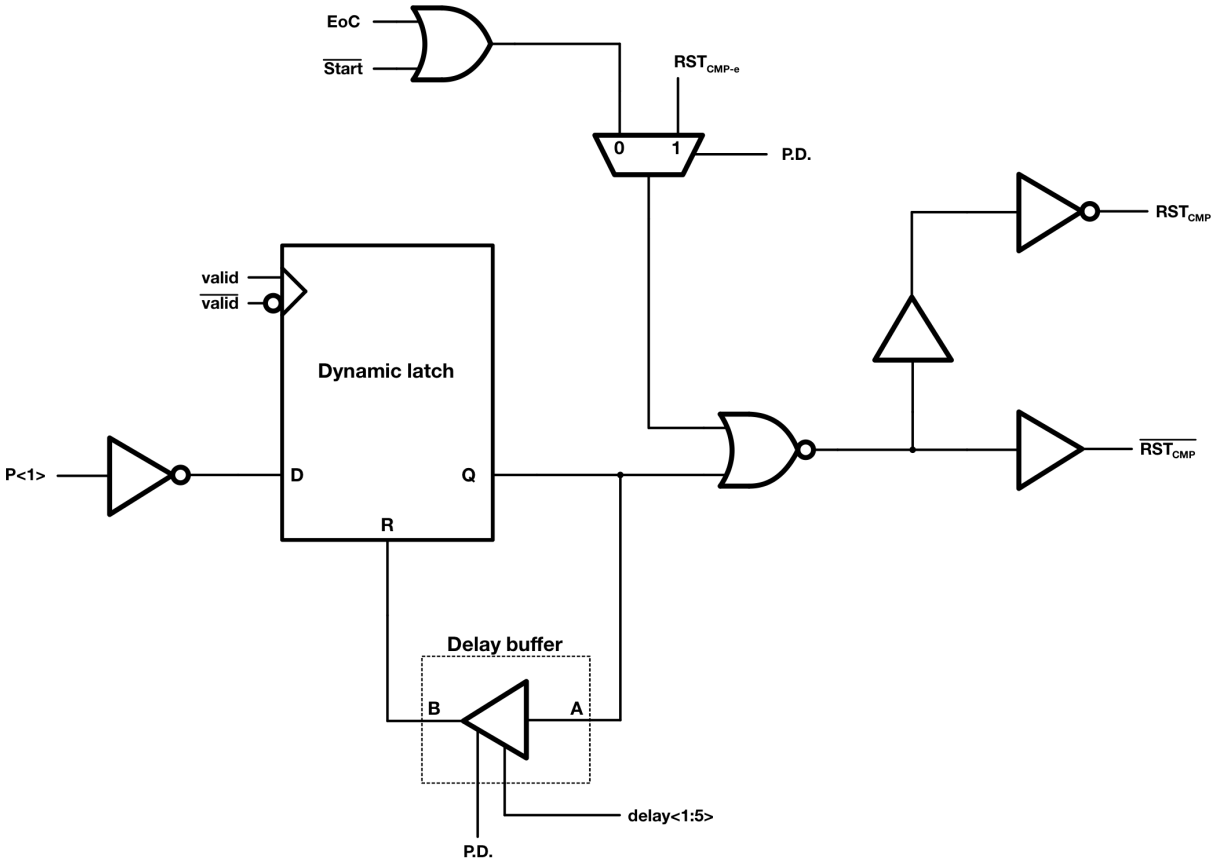

Figure 2.6: Schematic of the comparator reset loop.

The reset control loop is shown in Fig. 2.6. The *valid* signal is applied to the dynamic

latch as clock, setting its output to 1 when *P<1>* is pulled down. The reset time control is applied in the delay circuit, and so is the temporal power down. The remaining inputs, used for permanent reset of the comparator, are applied to the other gates.

The system is divided in two paths. In the first one, the OR gate is driven high when either *EoC* or $\overline{Start}$ is high. If *P.D.* is 0, this signal is propagated through the MUX to the NOR gate, which is driven low. This pulls RST_{CMP} up and $\overline{RST_{CMP}}$ down, resetting the comparator. This path also allows the beginning of the conversion: when the sampling phase is finished, $\overline{Start}$ is pulled down. Following the path, the OR gate is driven low, the NOR is driven high and the comparator reset is turned off, allowing it to operate. Thanks to the MUX, this path also resets the comparator if *P.D.* and $RST_{CMP\text{-}e}$ are 1.

The second path, consisting of the dynamic latch [6] and the delay buffer, is activated only after the first comparison is achieved, setting a 0 at the input of the NOR gate until *P<1>* is pulled down. When this happens, *valid* and $\overline{valid}$ make the DL transparent, so $\overline{P<1>}$ propagates to the NOR gate, driving it low and activating the reset. When the comparator is reset, its outputs are forced to V_{DD} and *valid* is pulled down, so the latch enters the hold mode. The next comparison is allowed thanks to the delay buffer: its high input is propagated to the asynchronous reset of the DL with a controllable delay, forcing the output of the latch back to ground. This turns off the comparator reset allowing the next comparison, but also propagates through the delay circuit and stops the reset of the latch so, when another comparison is completed, the latch is able to start the process again.

The temporal power down has an effect on this path too. If it is active, the output of the delay buffer is forced to 0. As a consequence, the latch is not reset and, if its output is pulled up, it keeps the input of the NOR gate at V_{DD}, keeping the comparator in reset state and stopping the conversion.

Another important feature of the circuit is the generation of the output signals. While $\overline{RST_{CMP}}$ is used to reset the first stage of the comparator (i.e., the preamplifier), RST_{CMP} is connected to the second one (the StrongArm comparator). Then, it is of vital importance that $\overline{RST_{CMP}}$ is pulled up some picoseconds before RST_{CMP} is pulled down, so when the second stage starts operating, the input value is the sampled input propagated through the preamplifier and not its reset voltage. That is why, to generate RST_{CMP}, the output of the NOR gate is buffered and then inverted instead of just inverted, This way, the delay of the chain is higher than the delay of the buffer that generates $\overline{RST_{CMP}}$. The buffer and the inverter also need to be big enough to drive the comparator, so the reset signals are pulled up and down in a reasonable time.

The main circuits in the system are the dynamic latch and the delay buffer, shown in Fig. 2.7 and Fig. 2.8, respectively. In the first one, the basic elements of a C^2MOS latch [6] are present: the tristate inverter, the reset transistor (in this case a PMOS) and the output inverter. It also has the same behavior: when *valid* and $\overline{valid}$ are 1 and 0, respectively, the input value is latched and, if the reset is off (R=0), propagated to the output through the inverter. The circuit also has a transmission gate controlled by the reset signal, to avoid cross-conduction current when the reset and *valid* are active at the same time. In addition, there is another transmission gate to feed back the output voltage, in the case where neither the reset nor the comparison done signals are active. In this case, the circuit works as a static latch, with two inverters creating a loop. When *valid* is pulled up, the feedback path is opened, avoiding cross-conduction current and longer propagation delay.

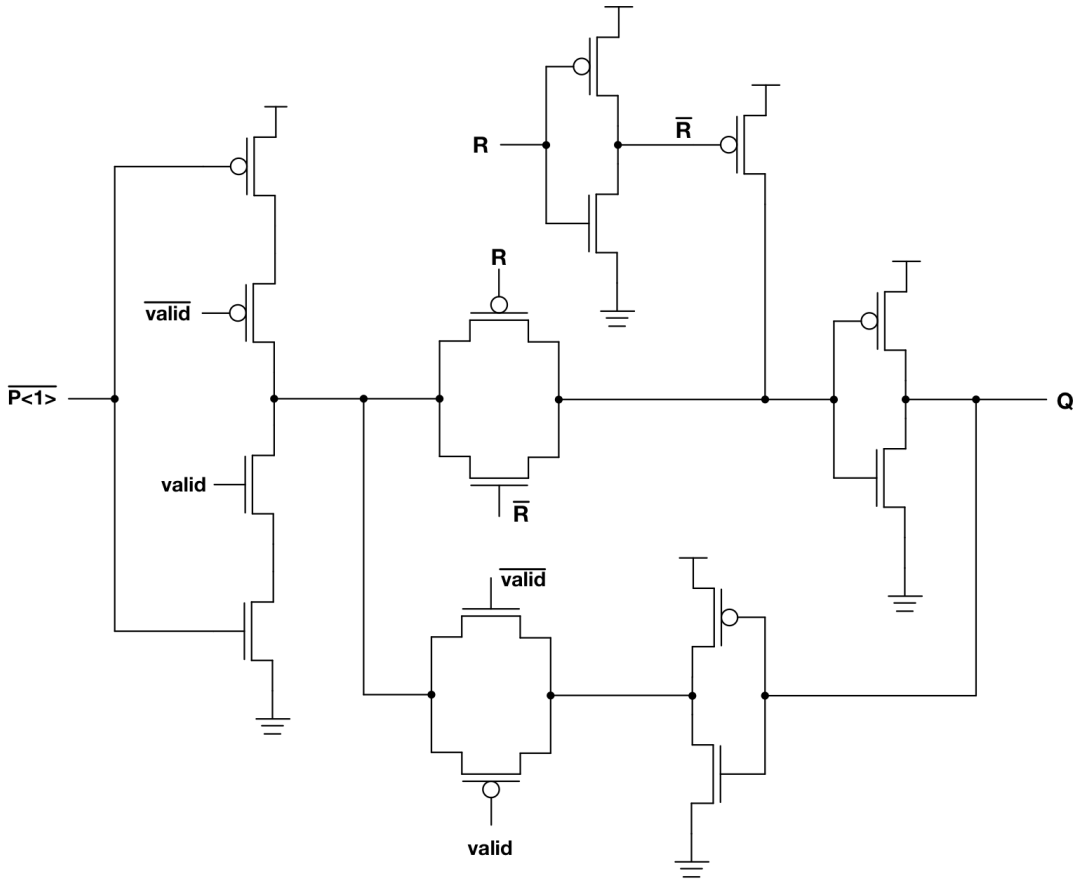

Figure 2.7: Architecture of the dynamic latch.

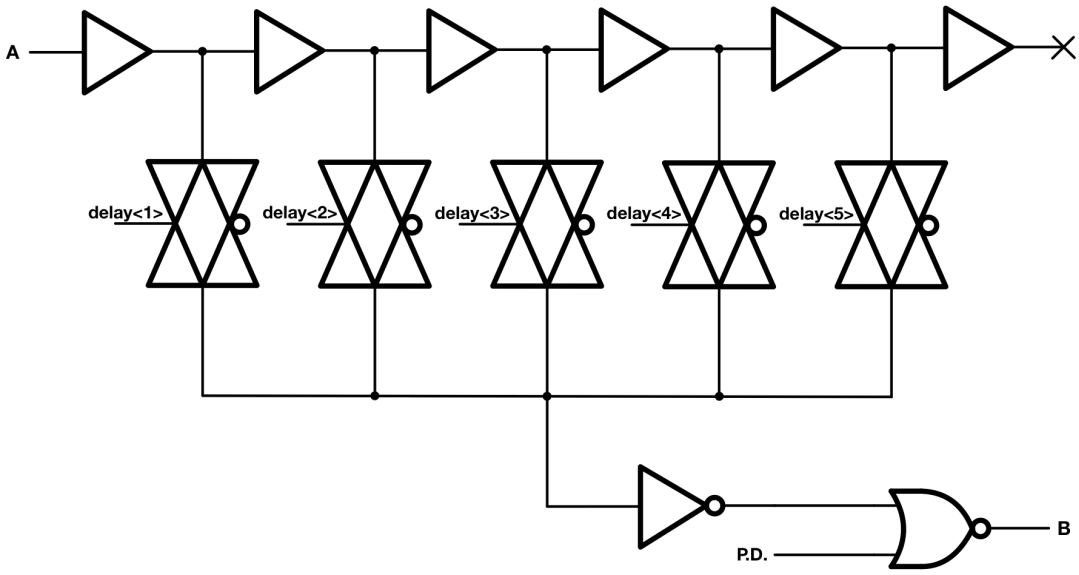

Figure 2.8: Architecture of the delay buffer.

The delay buffer exploits buffers and transmission gates. The selection signal decides which of the five buffer outputs is being propagated through the corresponding transmission gate. There is an extra buffer at the end of the chain so all the intermediate nodes are connected to the same load. The delay of the circuit is:

$$\boldsymbol{\tau} = i \cdot \boldsymbol{\tau}_{BUF} + \boldsymbol{\tau}_{TG} + \boldsymbol{\tau}_{INV} + \boldsymbol{\tau}_{NOR} \tag{2.1}$$

In the above formula, i is the selected delay, from 1 to 5, and represents the number of buffers through which the signal has propagated before arriving to the active transmission gate. Then, regardless of the select signal, the input propagates always through a transmission gate, the inverter and the NOR gate, which acts as another inverter if *P.D.* is 0, so the output is the delayed version of the input. Instead, if the temporal power down is on, the NOR gate is driven low and the output of the circuit is forced to ground.

2.3. Auxiliary circuits

Even though the two already explained circuits are the most important parts of the digital logic, they need support from other auxiliary circuits to generate different outputs and control signals, as the already mentioned "comparison done" signal. This circuitry can be divided in three subsystems: the end of conversion generator, the DAC equalization control and the reset mode control.

2.3.1. End of conversion generator

As already seen in the previous sections, the completion of the conversion is one of the conditions that stops the operation of the digital logic. A signal that indicates this condition is needed to keep the two main loops in reset state until the next sampling phase arrives. In addition, when the conversion is achieved the digital value has to be displayed at the output. This 13-bit value is generated by the DAC control loop and stored at the input of 13 C^{2}MOS dynamic flip-flops (identical to the ones used in the DAC control loop, showed in Fig. 2.3). These flip-flops have to be triggered as the conversion is finished.

To obtain the described signals, the following inputs are used:

- **Last comparison stored** ($CTRL_P$<13> and $CTRL_N$<13>): the conversion is not finished until the last comparison result is stored in the corresponding latch. Therefore, the trigger of the end of conversion generator is the differential output of the last latch, the signals $CTRL_P$<13> and $CTRL_N$<13>.
- **Sampling clock** ($\overline{CLK}$): the end of conversion signal has to be active until the start of the next sampling phase, triggered by the active low sampling clock.

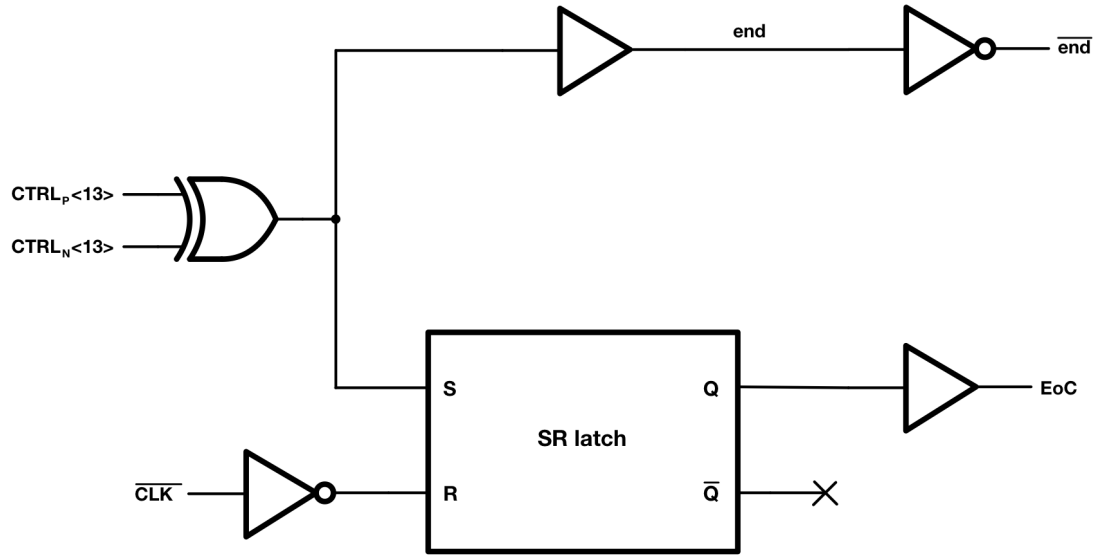

Figure 2.9: Schematic of the end of conversion generator.

Figure 2.9 shows the structure of the end of conversion generator. The XOR gate is driven high when either $CTRL_P$<13> or $CTRL_N$<13> is pulled down. The *end* and $\overline{end}$ signals, responsible of triggering the output DFFs, are pulled up and down respectively, and thus the digital value is displayed at the output. The XOR gate output also forces the SR

latch in "SET" mode, pulling up *EoC*.

As seen previously, the end of conversion is a condition to reset the DAC capacitors. This affects the end of conversion generator: when the DAC capacitors are reset, the inputs of the XOR gate are forced to V_{DD} and the XOR gate is driven low. Accordingly, the *end* and $\overline{end}$ signals are forced to 0 and 1, respectively. This means that these signals are showing a pulse waveform. Instead, the *EoC* output remains high until the SR latch is reset, which happens at the beginning of the next sampling phase thanks to the sampling clock ($\overline{CLK}$ is pulled down).

The SR latch architecture is shown in Fig. 2.10. Each of the inputs is connected to a NOR gate, whose output is connected to the input of the other one. When both inputs S and R are 0, both gates have the behavior of an inverter. Since the outputs are already the complementary version of each other, their values remain the same. When one of the inputs is pulled up, keeping the other low, the output of that gate is forced to ground. Then, this 0 is inverted in the other NOR, driving it high. It is not allowed to both set and reset at the same time (S=R=1), since the result would depend on which input remains high more time. The truth table is shown in Table 2.1.

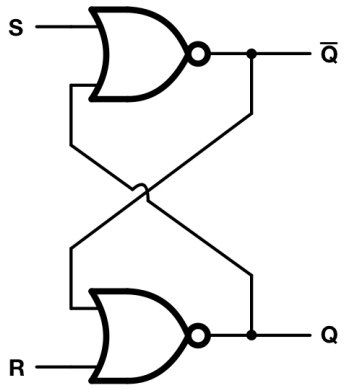

Figure 2.10: Architecture of the SR latch.

S	**R**	**Q**	$\overline{\mathbf{Q}}$	**Operation**
0	0	Q	$\overline{\text{Q}}$	Hold
1	0	1	0	Set
0	1	0	1	Reset
1	1	X	X	Invalid

Table 2.1: Truth table of the SR latch.

2.3.2. DAC equalization control

As already mentioned, when the conversion is finished, the DAC control signals $CTRL_P<i>$ and $CTRL_N<i>$ that have been pulled down during the conversion are forced back to 1. The DAC capacitors associated to these signals have their bottom plate voltage pulled from ground to V_{DD}, as when the conversion starts. Forcing all the capacitors bottom plates back to the same voltage they have at the beginning of the conversion phase means that the DAC is reset to the voltage it has when this phase starts, thus the sampled voltage. Keeping that voltage until the next sampling phase can lead to problems due to the difference between the input and the stored voltages: if it is too large, the settling time of the DAC voltage increases, and the sampled value might not be the correct one if the sampling period is too short.

To avoid this situation, the DAC is equalized. This means that the positive and negative halves of the capacitive arrays are connected through a switch after they are reset to the sampled values. Since the common voltage of the differential input is always $V_{DD}/2$, both nodes settle at this value until the next input is sampled. This assures the maximum variation the single-ended DAC can experience is $V_{DD}/2$.

The inputs needed for this behavior are the following:

- **Sampling clock** ($\overline{CLK}$): the equalization needs to stop before the beginning of the next sampling phase, so this signal has to stop the equalization when it is pulled down.
- **End of conversion** (EoC): the equalization starts after the conversion is achieved, so a signal with this information is needed. The EoC signal is 1 from the moment the conversion finishes until some picoseconds after the beginning of the next sampling phase. This is why the sampling clock is needed: if EoC is used to stop the equalization, for a short period of time the ADC would be sampling and equalizing at the same time.
- **DAC reset** ($\overline{end}$): even if the conversion is done, the equalization should not start until the DAC voltage is the sampled one: if it starts equalizing before, the common mode voltage it is not $V_{DD}/2$. As the DAC voltage after the equalization is the common mode voltage, the equalization should not start until the DAC reset is complete. The $\overline{end}$ signal is pulled up when the DAC control signals are forced back to 1 after the reset, when the equalization can start. Even though this signal is the real trigger of the circuit, it has other functions in the circuit that require that is high during the conversion, being pulled down when the last comparison is completed.

That is the reason why both EoC and $\overline{end}$ are necessary: the equalization can start only if both signals are high.

Figure 2.11 shows the architecture of the DAC equalization control. When the conversion is achieved, EoC is pulled up and $\overline{end}$ is pulled down, so the AND gate is still driven low. Then, the DAC is reset and the $\overline{end}$ signal is pulled up again, driving high the AND gate. If $\overline{CLK}$ is high, i.e., the ADC is still in the conversion phase, the NAND is driven low, and the output signals equ and $\overline{equ}$ are pulled up and down respectively, which turns on the aforementioned equalization switch.

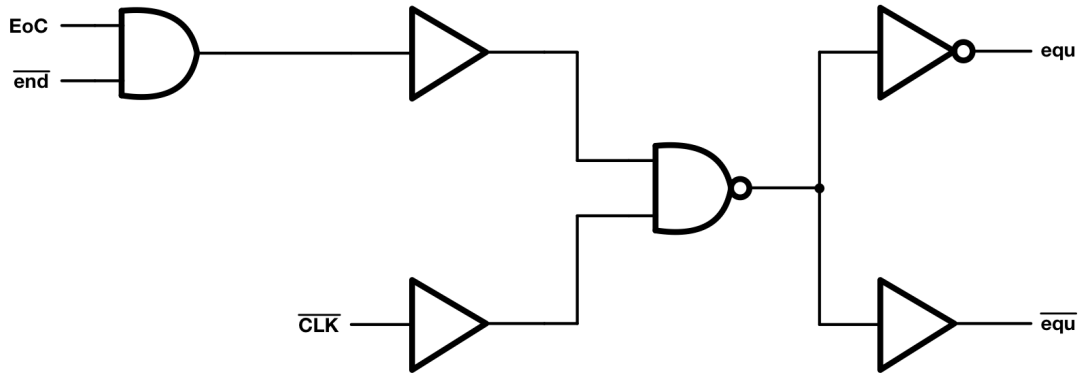

Figure 2.11: Architecture of the DAC equalization control circuit.

2.3.3. Reset mode control

Although the logic is designed to work at high speed, the converter clock frequency depends on the specific application. In the case of low conversion rate, the actual conversion time represents just a small percentage of the overall conversion period. In that case, the logic must enter a reset mode, where all the other blocks are turned off until the next conversion starts. This reset mode should be also adopted in the sampling phase of the converter and when the external reset is active.

The reset mode control is responsible for generating the signals that turn off (and eventually also on) the different circuits of the logic. It is triggered by three different inputs:

- **The sampling clock** ($\overline{CLK}$): since the logic must be off when the converter is in the sampling phase, this input is needed. It is low during the sampling phase, while it is pulled up when the conversion starts.
- **The external reset** ($\overline{reset}$): if something goes wrong during the conversion, this active low signal can be controlled from the outside, resetting the logic. In the case of normal behavior, it is high during the whole process.

- **The end of conversion** (EoC): when the conversion is finished, this signal is set to 1 until the next sampling phase begins.

While the first two inputs have the same effect on the circuit, EoC is just responsible of the reset of the DAC. As already explained, this does not mean that the comparator is not sent into the reset state, but that this feature is already included in the comparator reset loop. Instead, $\overline{CLK}$ and $\overline{reset}$ take care of the reset of all the bistable elements in the DAC loop and keep the comparator in reset state.

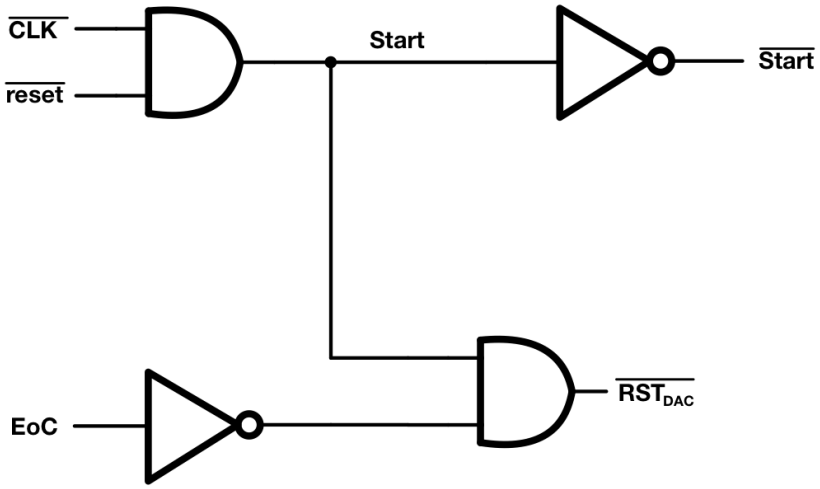

Figure 2.12: Architecture of the reset mode control circuit.

The design of the controller is shown in Fig. 2.12. When either $\overline{CLK}$ or $\overline{reset}$ goes to 0, the first AND gate is driven low, thus $Start$ is pulled down and $\overline{Start}$ is pulled up. The second AND is driven low, activating the DAC reset (i.e., $\overline{RST_{DAC}}$=0). This last output can also be forced to 0 due to EoC, resetting the DAC when the conversion is completed.

2.4. Global behavior

With every subsystem of the circuit already analysed, the next step must be a complete description of the whole digital logic. A conversion starts with the sampling phase, during which the reset mode control sets $Start$=0, that keeps all the shift-register of flip-flops of the DAC control loop in reset state, and $\overline{Start}$=1, which resets the comparator through the comparator reset loop.

When the conversion phase starts, $\overline{CLK}$ is forced to 1, pulling $Start$ up and $\overline{Start}$ down. While the DFFs in the shift-register keep the same output forced by the set, $\overline{Start}$ propagates through the comparator reset loop and turns off the reset, allowing the first comparison. When it is completed, the comparison done signals are activated (i.e., *valid* is pulled

up and $\overline{valid}$ is pulled down), triggering both the DAC control loop and the comparator reset loop.

In the first loop, all the flip-flops are triggered but, since they are all in reset state, all their outputs are high. As they are connected in a shift-register, the input of all of them is at V_{DD}, excepting the first one whose input is always connected to ground. Thus, while all the other propagation signals remain high, $P<1>$ is pulled down. This triggers the first latch, storing the result of the comparison and changing the corresponding switch of the DAC thanks to the differential output $CTRL_P<1>$ and $CTRL_N<1>$. It also sets to 0 the input of the second DFF, allowing that, after the next comparison (i.e., $valid$=1), $P<2>$ goes to 0, enabling the second latch to store the corresponding bit.

In the second loop, the dynamic latch is set in transparent mode. Few instants later, $P<1>$ is forced to ground, so the input of the latch switches to V_{DD}, staying high for the remainder of the conversion process. As the latch is transparent, this 1 is propagated to its output, triggering the reset of the comparator. After some time, the comparison result signals CMP_P and CMP_N are forced to 1, resetting $valid$ to 0 and $\overline{valid}$ to 1, and keeping the DL in hold mode. In addition, the delay buffer is activated due to the change in the dynamic latch output. After the selected delay, the reset of the latch is pulled up, so its output is pulled down and the comparator is allowed to make the second operation. The DL is kept in the reset state until the 0 is propagated through the delay buffer, but this occurs before the next comparison is finished, so it does not affect the behavior of the system.

The process is repeated 13 times: the 0 is propagated through the flip-flops, the latches store the comparison result and change the DAC switches, the dynamic latch in the reset loop is made transparent, allowing the comparator reset, and the delay buffer is activated, resetting the DL and allowing the next comparison. After the last iteration, the differential output of the last latch ($CTRL_P<13>$ and $CTRL_N<13>$) starts the end of conversion generator. The end and $\overline{end}$ signals trigger the 13 flip-flops that display the digital result at the output of the circuit. EoC is also set to 1. Due to this, the comparator is reset, preventing unnecessary operations. In the reset mode control circuit, $\overline{RST_{DAC}}$ is pulled down, forcing all the latches outputs to 1. This means that the XOR gate at the input of the end of conversion generator is driven low, pulling end down and $\overline{end}$ up. When this occurs, the equalization is allowed and the process finished.

2.5. Simulation results

After the description of the logic, its time to analyze its simulated behavior. To avoid simulating the whole ADC, a constant input voltage of 50mV is connected to the comparator. Then, the only needed stages are the comparator and the digital logic. The simulation is done in these conditions: standard model of the transistors, a temperature of 100º C, intermediate comparator reset time ($delay<3>$ set to V_{DD}) and conversion frequency set to 250MHz. Under these conditions, the main signals are shown in Fig. 2.13 (a DAC control signal, $CTRL_{P,N}<2>$ is shown as an example). The priority of the different signals can be observed: before $\overline{Start}$ is pulled down the system is in reset state; *valid* is pulled up after either CMP_P or CMP_N is pulled down. Then, the correspondent DAC control signal is pulled down before the comparator is reset. After this, both comparator outputs are reset to V_{DD} again, pulling down *valid*. The next comparison can start when the comparator reset is turned off. After the last comparison, *EoC* is pulled up, forcing all the DAC control signals to 1. When this happens, the equalization is activated. Finally, the conversion period ends and $\overline{Start}$ is pulled up, forcing all the signals to their original value.

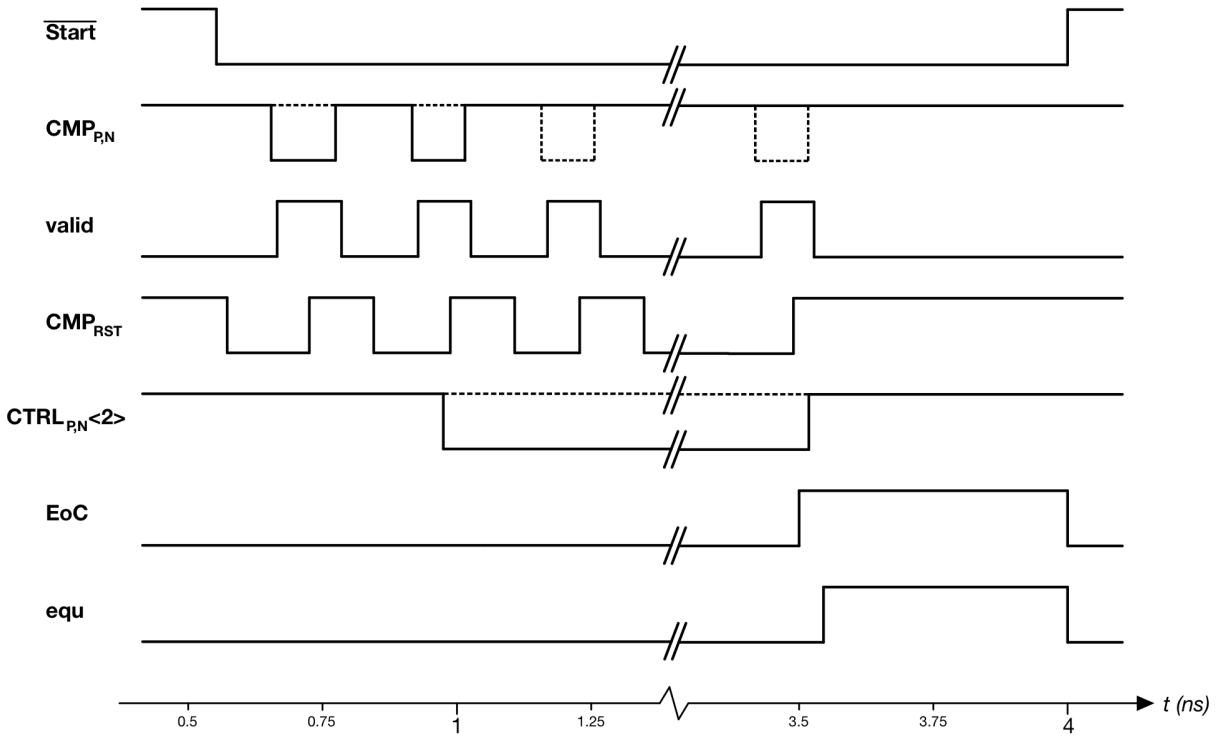

Figure 2.13: Waveform of the main signals of the existing logic.

DAC control loop	
Logic delay	DAC settling
57.55 ps	136.15ps
Comparator reset loop	
Logic delay	Comparator reset
86.41 ps	107.29 ps

Table 2.2: Delay on a loop step in the existing logic.

Table 2.2 shows the delay for a single comparison step in each loop. Considering also the comparator delay (40ps per comparison), for each step of the conversion the delay is 233.7 ps. Then, the delay for the 13 comparison steps is 3.038 ns. As the conversion phase takes 7/8 of the total conversion time because of the interleaved operation of the converter, we can approximate the latter parameter to 3.472 ns, without considering the initial and final operations. Since the goal of this logic is to achieve a conversion frequency of 2.5 GHz, the 8 ADCs should operate at 312.5 MHz, i.e., the conversion time should be 3.2 ns. The achieved delay, before even considering parasitics, is already higher than the target one. This is due to two limitations:

- **DAC settling time**: the 136.15 ps the DAC has to settle takes almost a 60% of a bit evaluation time, but the DAC actually needs just 60 ps to settle. Thus, this time needs to be reduced. To achieve this, the temporizer needs to be modified, since is the circuit that controls both the DAC settling and the comparator reset times. The controllable delay of this circuit can't be used to reduce the DAC settling time, since its goal is to counteract the variations due to PVT. It is important to mention that the temporizer delay can be reduced until reaching the minimum settling time of 60 ps, while the comparator reset is not a limiting factor.
- **Logic delay**: the system can benefit from reductions in the delay of both the DAC control loop and the comparator reset loop, increasing the maximum sampling frequency while keeping the same DAC settling and comparator reset time.

Considering these limitations, it is clear that an optimization is needed to reduce the delay of both loops and the temporizer, with the goal to reach the minimum DAC settling time.

In terms of power consumption, Fig. 2.14 shows the variation of power consumption with frequency. There is a linear relation until 275MHz, while for higher frequencies the conversion is not achieved.

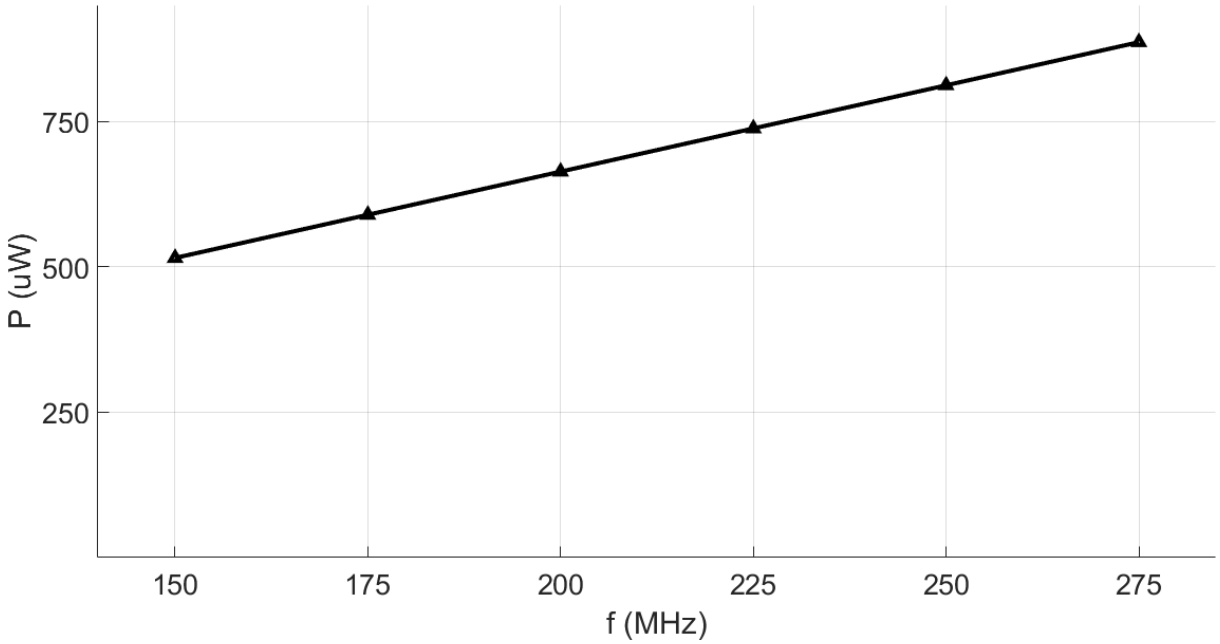

Figure 2.14: Power consumption for different frequencies.

Chapter 3

Proposed SAR logic

The logic presented in the last chapter is too slow to achieve the expected conversion rate of 312.5MHz. While optimization of all the subsystems is important, this thesis focuses on improving the two main loops of the system. The reason for this is that a delay reduction on a loop is multiplied by the number of steps in the process, making it more relevant than an improvement in a circuit that is just activated once.

Another problem in the original logic is that the possibility of a metastable event is not considered. If that situation occurs, the conversion could not be completed. To avoid it, two architectures for a metastability detector are designed and compared.

Lastly, a new subsystem is designed: a circuit that controls the comparator noise. The comparator is equipped with a controllable capacitance between the preamplifier and the latch, with the goal of increasing it after an specific comparator step, thus reducing the latch input referred noise. The designed circuit connects this capacitance before the next comparison, with time enough to settle before the comparator is turned on.

3.1. DAC control loop

3.1.1. New architecture design

The improvements made in the DAC control loop start applying the ideas presented in [7]. Figure 3.1 shows the devised architecture. The main advantage of this topology is that the logic manages to bypass the delay of the flip-flop in the path between the comparator outputs (CMP_P and CMP_N) and the DAC control signals ($CTRL_P$<1:13> and $CTRL_N$<1:13>). Therefore, the delay of the logic in a comparison step is just the delay of a latch. To achieve this, the propagation signals P<1:13> are used to generate enable signals that allow the operation of the next latch, i.e., P<i> generates E<$i+1$>.

When a comparison is achieved, the result is stored in a latch that is already enabled and doesn't have to wait until it is triggered by the shift-register. The first DL doesn't have an enable signal, so it is triggered by *P<1>* as in the existing logic.

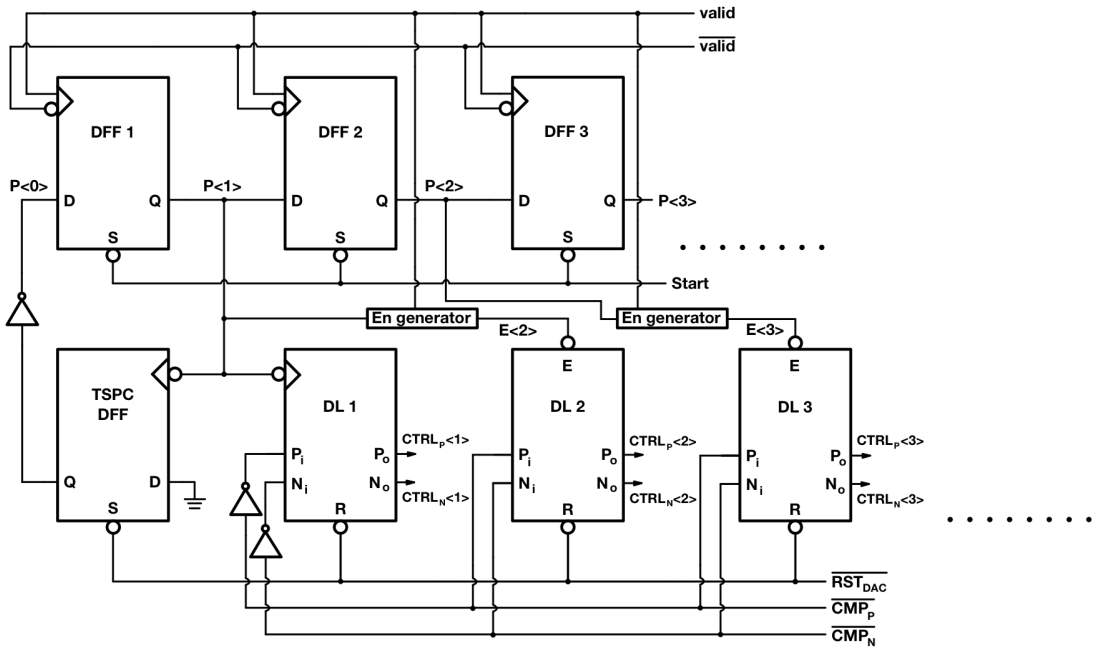

Figure 3.1: Architecture of Cao's DAC control loop.

The propagation signals *P<1:13>* are now used to enable the dynamic latches thanks to the enable generator. Due to this, now the signals have a pulse waveform. This can be achieved adding a TSPC flip-flop [6] that is initialised to 1 and an inverter. The flip-flop is triggered by the falling edge of DFF1 output, *P<1>*, while its input is connected to ground. Its output is inverted and connected to the input of the first flip-flop *P<0>* (which was connected to ground in the existing logic).

At the first step of the conversion, *P<0>* is low. When the first comparison is done, this 0 propagates through the first DFF and triggers the TSPC flip-flop. This sets *P<0>* to 1. In this step, also the first latch DL1 is triggered by *P<1>*, being the only latch whose behavior and architecture is the same as in the existing logic, shown in Fig. 2.4. At this moment, DL2 is enabled by *P<1>*. When the second comparison is completed, the comparator output is stored in DL2. Then, the 0 at *P<1>* propagates to *P<2>*, while *P<1>* goes back to V_{DD}. Thus, DL3 in now enabled while the enable signal of DL2 is pulled up, turning off the latch. The 1 in *P<1>* propagates also through the shift-register, so the 0 remains only for one pulse in each *P<i>* signal. This way, just one DL is enabled at a time.

The latch structure has to be now modified in order to include an enable input and start working when the comparison is achieved, instead of being clocked. A stage with this behavior is designed in [8]. The adapted version of the circuit for the designed logic is shown in Figure 3.2.

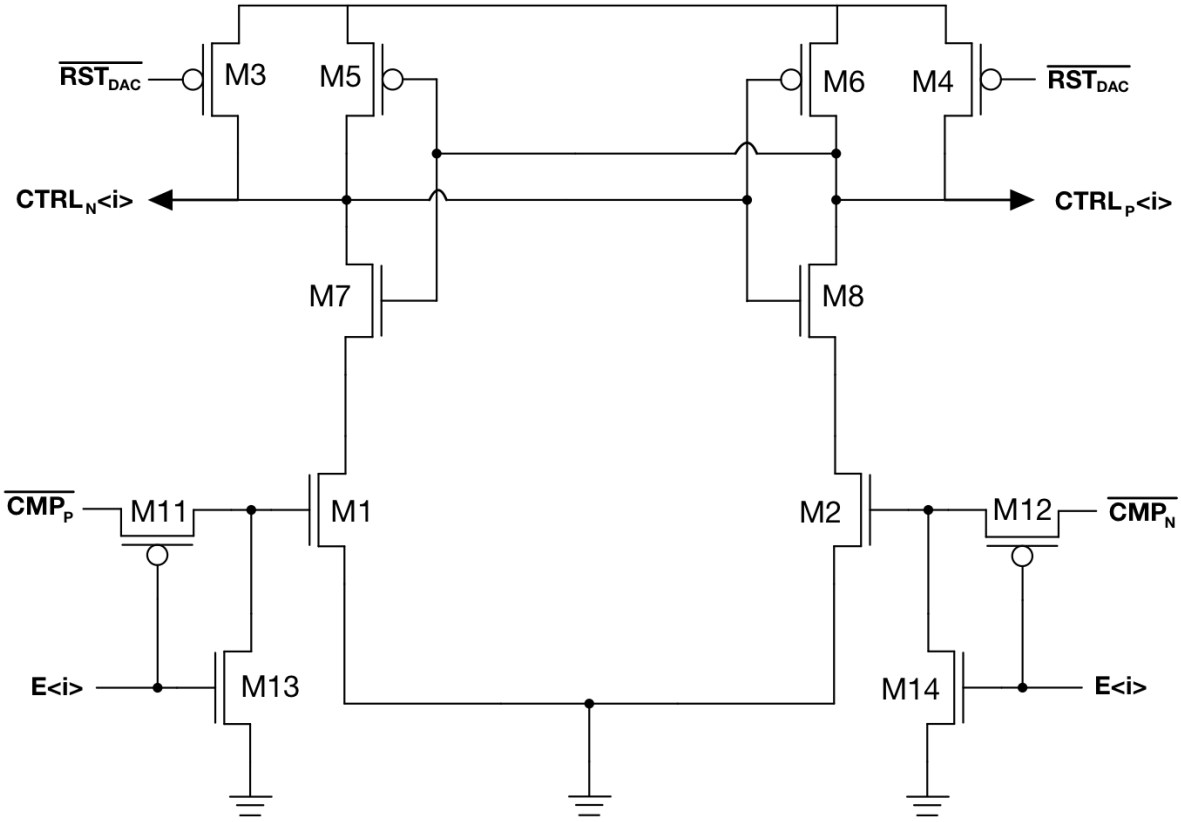

Figure 3.2: Architecture of the latch with enable.

There are some important differences between this latch and the one used in the existing logic, shown in Fig. 2.4. First, the tail transistor is removed, so the DL is not clocked by an external signal. The trigger is instead the comparator output that is pulled down after the comparison. This is also the reason for the second difference: the inputs of the latch are the inverted outputs of the comparator. If the comparator outputs were directly applied, M1 and M2 would be active and the outputs of the latch would discharge to ground. Instead, now M1 and M2 are off until the comparison is finished. At this moment, one of the inputs is pulled up, discharging the corresponding DL output. As opposed to the original logic, now the output of the comparator and the corresponding DAC control signal are connected to the opposite ends of the latch, i.e., while $\overline{CMP_P}$ is connected to the left half latch, $CTRL_P<i>$ is connected to the right. The reason for this is that in the case of a positive comparison, $CTRL_P<i>$ must be pulled down. While in the original

latch architecture (Fig. 2.4) CMP_P was able to force the output node to 0, now is $\overline{CMP_N}$ the input that pulls down the half latch in the case of a positive comparison result.

The third difference is the elimination of M9 and M10, the NMOS transistors that help discharging the output nodes in the original latch (Fig. 2.4) when it is clocked, ensuring that there is always a low impedance path to ground. In this new architecture without the tail transistor, this path would discharge the output nodes right after the reset is turned off, so it can't be used and the output nodes need to be a high impedance at some points of the conversion. Also, the latch is not reset anymore by the propagation signal (since this would reset the DL when $P<i>$ is pulled up again, losing the comparison result), but just by the DAC reset. This means that, since the start of the conversion and until the latch is triggered, their outputs are high impedance nodes.

The last difference is the implementation of the enable, done with M11-M12 and M13-M14. When the enable is active (i.e., low), M11-M12 are turned on and the input signals are applied to the gate of M1-M2. If the enable is high instead, M13-M14 are on and the gate of M1-M2 is set to ground, turning them off. The pulse waveform of the enable signal ensures that only one DL is active at a time. Consequently, each comparator output is connected to one NMOS gate and twelve PMOS drains instead of thirteen NMOS gates. Since the PMOS are smaller, the load at the comparator output is reduced.

Although the propagate pulse signal $P<i>$ might seem directly applicable as the enable for the next step $E<i+1>$, this is not the case. $E<i+1>$ cannot be pulled down until the comparator is reset, or the result of the i-th comparison would be stored in DL(i+1). Thus, a circuit to generate the enable signal is required. Figure 3.3 shows the implemented solution.

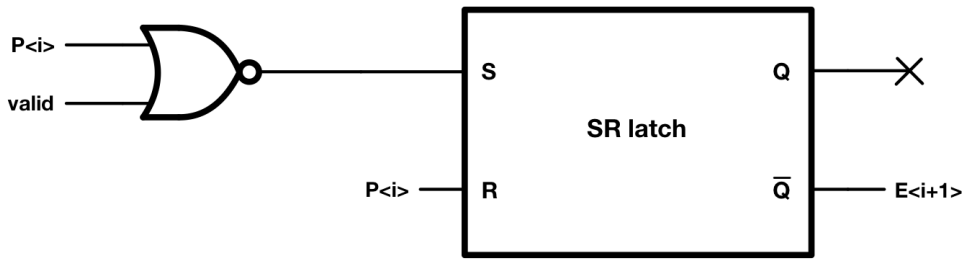

Figure 3.3: Generation of the latch enable signal.

The SR latch (Fig. 2.10) is in reset state when $P<i>$ is 1. When the pulse arrives, it stays in hold mode (S=0, R=0) until the comparator is reset and *valid* is pulled down. At this moment, the NOR gate is driven high. This puts the SRL in set mode, activating $E<i+1>$. When the next comparison is completed, *valid* is pulled up driving low the

NOR gate and putting the SR latch in hold mode (S=0, R=0). After that, *P<i>* is pulled up resetting the SRL and pulling up *E<i+1>*. As it can be seen in Fig. 3.3, the negative output of the SRL is used as the enable, since it is an active low signal.

3.1.2. Problems with the new architecture and final design

While the proposed structure has important advantages regarding the DAC control loop delay, it loses an important feature that appears in the original design: the generation of step signals that represent the completion of a conversion step. These signals are necessary to debug the system in case of malfunction and they are used in the comparator noise control, explained in subsection 3.3.1.

Another issue that can be seen in the simulations is the generation of leakage currents in the latches of Fig. 3.2. Even though M1 and M2 are off, a current is flowing through them. Since at the beginning of the conversion phase M7-M8 are already on, the output nodes, which are at high impedance, start discharging. This is not a problem for the first latches, since they are triggered few moments after the start of the conversion, so the leakage is harmless. Instead, in the last ones (DL10-DL13) the discharge can activate M5 or M6, forcing one of the outputs to 1 while the other one keeps being discharged, storing an incorrect value. Leakage also occurs in the original latches (Fig. 2.4), but the output nodes are connected to V_{DD} through M3-M4 until the propagation signal that triggers the DL is pulled down (see Fig. 2.4).

Lastly, 12 enable signals need to be generated, which means adding 12 SR latches. Also, the *valid* signal is connected to 12 XOR gates, increasing its load. This makes the signal slower and increases power consumption, because this load has to be charged 13 times during the conversion, one for each step.

Even though the first issue can be solved using 13 flip-flops triggered by the propagation signals *P<1:13>*, the increase in area and power consumption makes it a non-optimal solution. Some circuitry could also be added to avoid the leakage, but in order to keep the delay of the stage at the minimum value, the optimal solution is to go back to the step waveform in the propagation signals *P<i>* and trying to adapt it to the use of latches with enable. This way, the first and second problems, which don't appear in the existing logic, can't either appear in this new architecture.

The structure of the modified loop is shown in Fig. 3.4. The line of flip-flops is identical to the original, but now *P<i>* is responsible of generating the enable of the DLs. The only exception is the first latch, which is still triggered by *P<1>*. Since the propagation signal is not a pulse anymore, the TSPC flip-flop is not necessary.

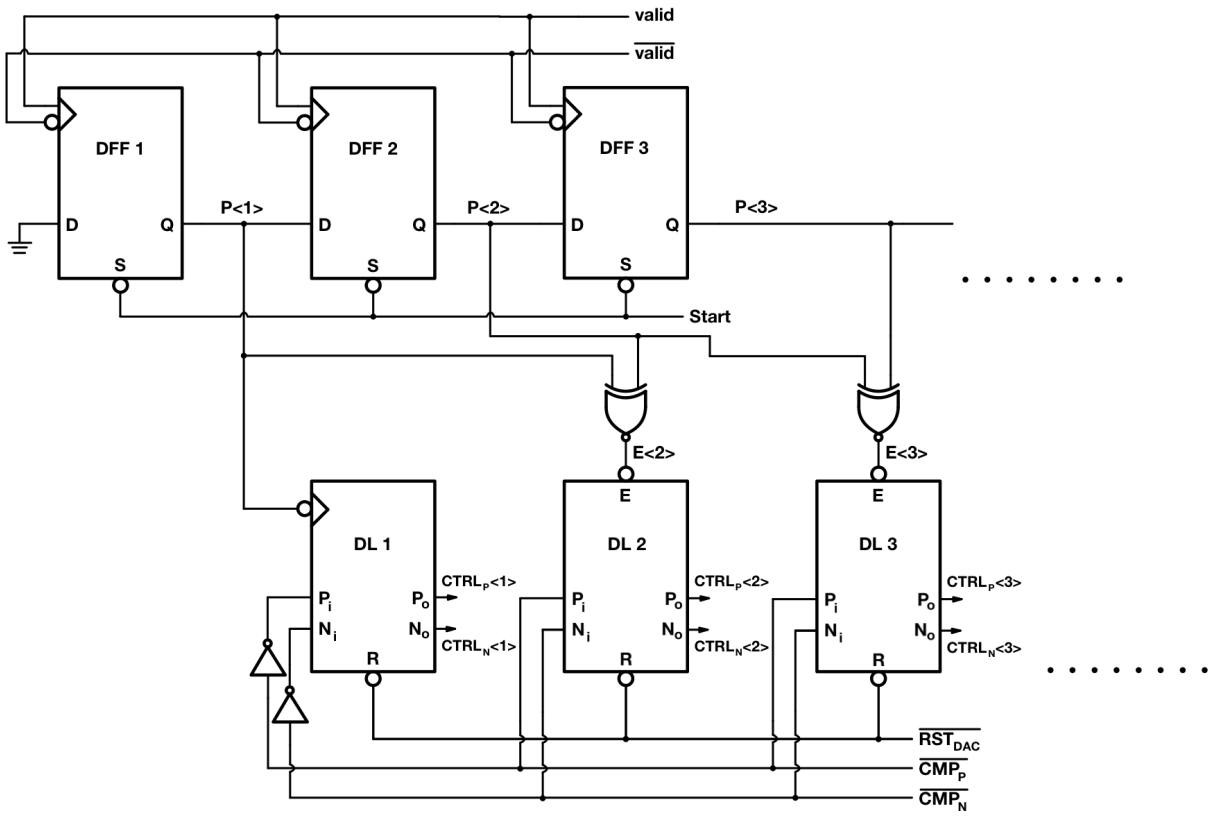

Figure 3.4: Architecture of the first modified DAC control loop.

The enable signal $E<i>$ is now generated using the propagation signal corresponding to the previous comparison, $P<i\text{-}1>$, and the one corresponding to the current, $P<i>$, as inputs of an XNOR gate. This generates a pulse signal that is 0 (so the latch is enabled) when the comparison is achieved.

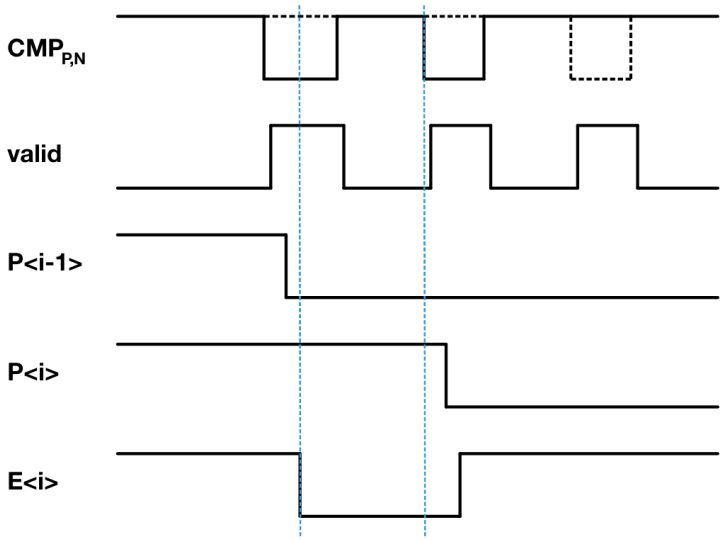

Figure 3.5: Involved signals in the generation of $E<i>$

Figure 3.5 shows the downsides of this solution. First, a DL can be enabled when the comparator has not been reset, thus the bit evaluation is stored also in this latch. Second, the time window between the comparison decision and the pull up of $E<i>$ might be tight, since it is just the delay in the pull up of *valid*, a DFF and an XNOR gate.

The first problem can be avoided using an enable generator circuit, like the one shown in Fig. 3.3, to make sure that the latch is enabled after *valid* is pulled down. However, adding extra logic gates to the circuit should be avoided. The second drawback might not cause malfunction if the DLs are fast enough, but if it is not, the system would not work. Consequently, the structure of the shift-register needs to be modified again.

The origin of the weaknesses is that $P<i>$ is generated before the comparator is reset. Therefore, a new structure is needed, where $P<i>$ is forced to 0 after *valid* is pulled down. However, the different behavior of the first DL complicates this solution, because if $P<1>$ is pulled down after the comparator is reset, when the first latch is triggered, there is no value at the comparator to store. Thus, a modification of the circuit is needed so this latch has the same behavior as the others. This also benefits the conversion speed, since the first comparison step is the slowest, as it has to wait for the delay of both the DFF and the DL. Figure 3.6 shows the modifications implemented to solve these problems.

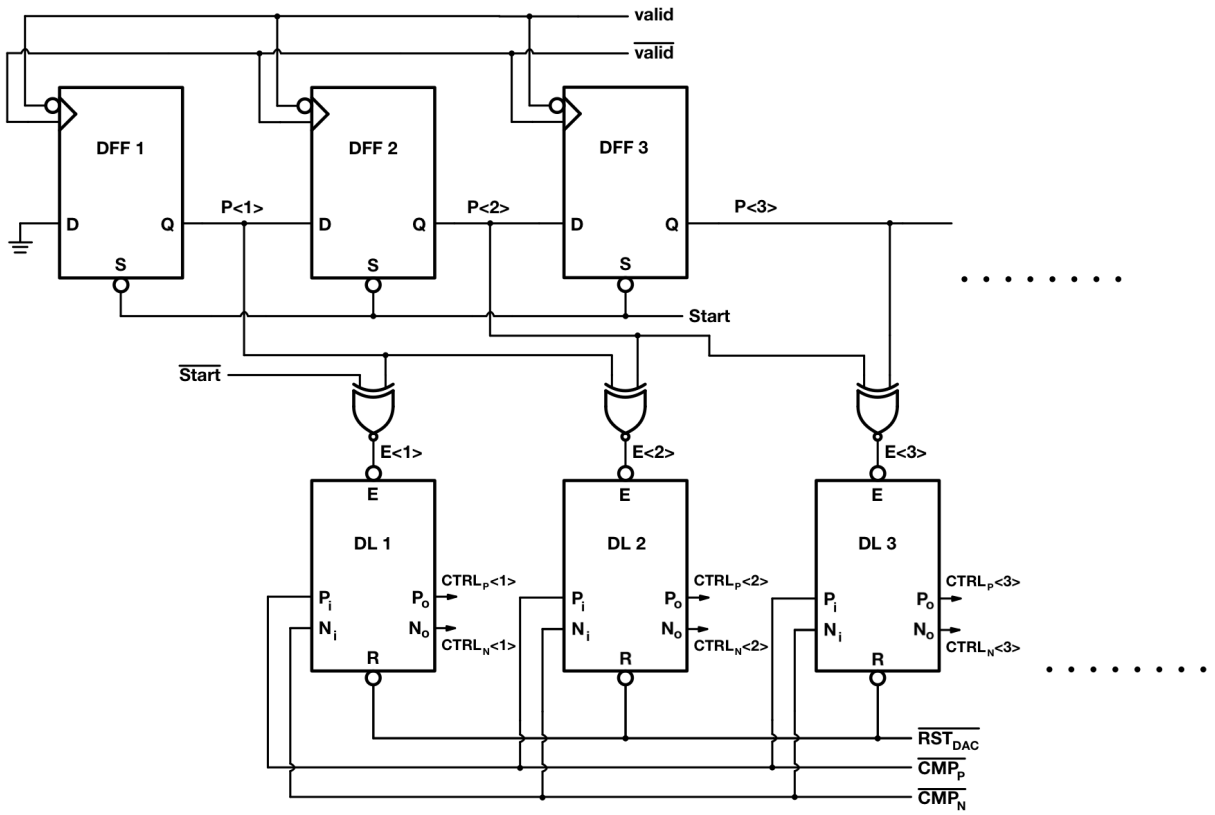

Figure 3.6: Final architecture of the DAC control loop.

Thanks to the low-edge-triggered DFFs, now $P<i>$ is always pulled down after *valid* is forced to 0. Generating the enable signals with an XNOR gate ensures that the comparator is reset before the DLs are enabled. Also, the first latch is now identical to the others. To generate $E<1>$, the sampling clock $\overline{Start}$ is used, thus, if the external reset ($\overline{reset}$) is 1, the latch is enabled since the beginning of the conversion phase and until the comparator is reset after the first comparison, so the result of this comparison can be stored. Another advantage of this architecture is that now *valid* is neither connected to the DLs nor used to generate $E<i>$, reducing its load.

The low-edge-triggered DFFs are again designed in a C^2MOS topology [6], as shown in Fig. 3.7. The desired behavior is achieved interchanging *valid* and $\overline{valid}$ in the tristate inverters. Also, the reset is now implemented at the node between the two tristate inverters (node A) rather than at the input of the last inverter (node B). The reason for this is that *valid* and $\overline{valid}$ are 0 and 1 respectively, during and right after the sampling phase. If a 0 from the previous comparison is present at node A, it needs to be set, or when the conversion phase starts, it can propagate through the second tristate inverter to node B and then to the output, pulling down all the propagation signals $P<i>$. This reset configuration doesn't need an inverter as in the existing logic, so 13 less inverters are needed, reducing area and power consumption.

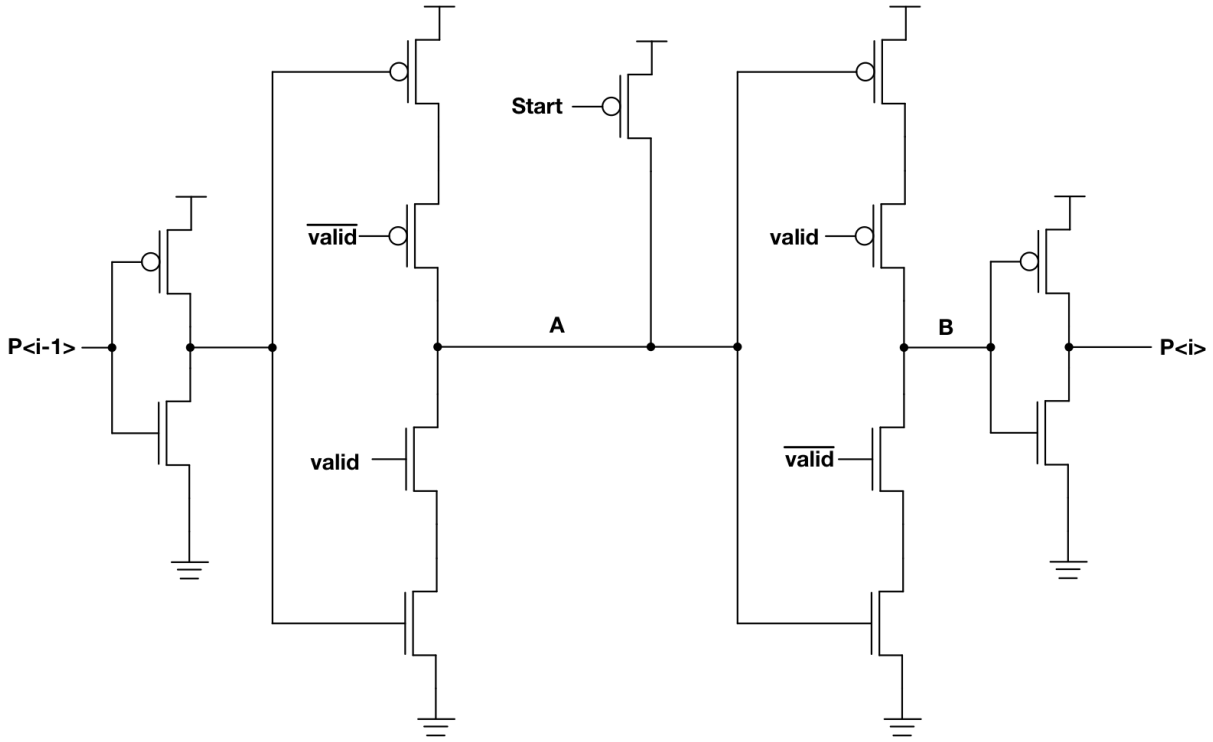

Figure 3.7: Architecture of a low-edge-triggered DFF.

Figure 3.8 shows the main signals of the final architecture. All the enable signals now are activated when the comparator is reset, and they are not pulled up until the comparison step is completed, giving enough time for the latches to store the comparison result.

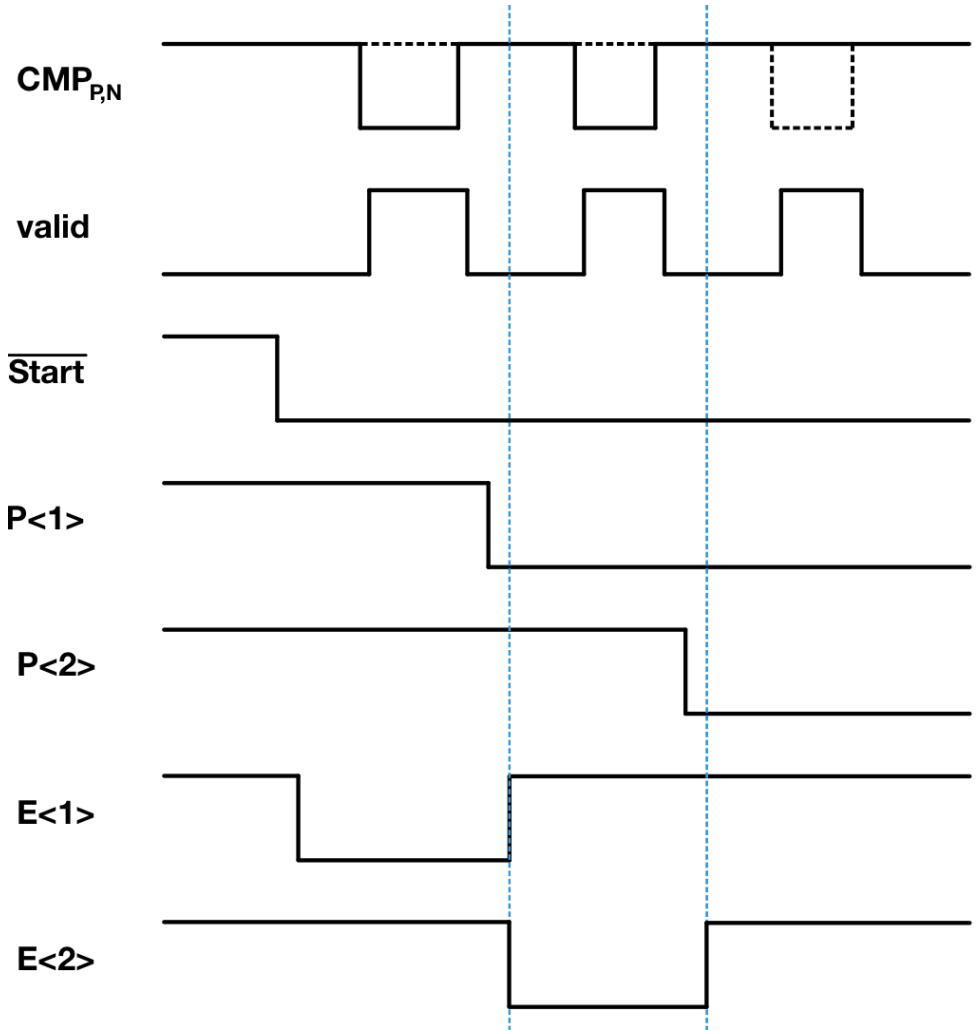

Figure 3.8: Final architecture signals waveform.

Finally, to complete the optimization of the DAC control loop, the AND gates for the external control of the DAC described at the end of Section 3.1 are removed. Even though this feature is lost, now the path has less gates, reducing delay, area and power consumption.

3.2. Comparator reset loop

With the modifications applied to the DAC control loop, the delay of the temporizer can be reduced keeping the same DAC settling time. Moreover, the a new DAC with reduced

settling time is used in the new version of the converter, so the temporizer delay can be further reduced. However, these improvements suppose that the comparator loop is now the limitation of the logic in terms of speed. To avoid this, a new architecture is proposed for the comparator reset loop [9], depicted in Fig. 3.9.

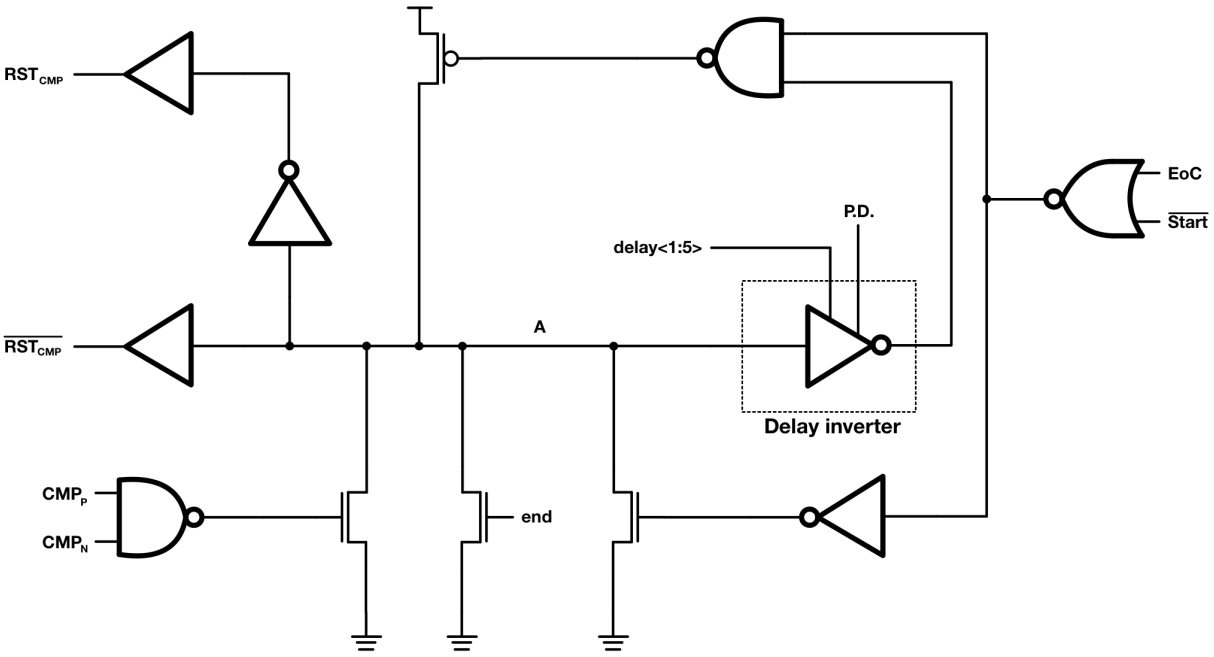

Figure 3.9: Architecture of the proposed comparator reset loop.

The main advantage of this configuration is that the path between the comparator outputs CMP_P - CMP_N and the reset signals RST_{CMP} - $\overline{RST_{CMP}}$ has fewer stages: instead of the generation of *valid* and $\overline{valid}$, the dynamic latch, a NOR gate and the output buffers/inverters, as shown in Fig. 2.6, now the path consists of a NAND gate driving an NMOS transistor, which pulls down the input of similar output buffers/inverters (i.e., node A). The path to stop the reset of the comparator is also different, but with a similar number of stages: while in the existing logic it is made of the delay buffer, the dynamic latch, the NOR gate and the buffers and inverters, in the new structure it consists of a delay inverter and a NAND gate driving a PMOS transistor, which pulls up the node A. This path might have a higher delay than the original one, but that difference can be reduced modifying the temporizer, which has a similar behavior of the delay buffer, i.e., its delay can be trimmed and it can be stopped by the temporal power down signal *P.D.*

Besides the normal operation of the loop, the circuit needs also to reset the comparator if the converter is sampling, if the external reset is active or if the conversion is finished. $\overline{Start}$ is 1 in the first two situations, while *EoC* is pulled up in the last case. Both signals

are used as inputs of a NOR gate. If any of this conditions is met, the gate is driven low, the NAND gate connected to its output is driven high and the PMOS is turned off, being unable to pull up the node A. However, this is not enough, since the node might already be at V_{DD}: it needs to be pulled down. For that reason, the output of the NOR gate (which is 0 under any of the aforementioned conditions) is inverted and connected to the gate of an NMOS transistor that pulls down the node A, activating the comparator reset.

This path also starts the conversion cycle after the sampling phase. The node A is pulled down when $\overline{Start}$ high (i.e. during the sampling phase), so the output of the delay inverter is 1. Then, at the beginning of the conversion phase, $\overline{Start}$ is pulled down and the NOR gate is driven high. As both inputs of the NAND gate are 1, it is driven low, turning on the PMOS and pulling up node A, allowing the comparator to start its operation.

It is important to take into account that this logic is ratioed, i.e., the behavior of the circuit depends on the aspect ratio of the different transistors. If may happen that the delay inverter is not properly trimmed, so the 0 in the node A during the reset of the comparator propagates to the PMOS, turning it on, while the comparator is not still completely reset. In that case, the input NAND is driven high until the comparator is finally reset, thus, both the NMOS and the PMOS are active. Since the reset state must be kept until the NAND gate is driven low, the NMOS must be bigger than the PMOS. Another similar case can be considered: the delay inverter is too slow and a 1 in the node A is not propagated to the gate of the PMOS before the conversion is achieved and the NMOS is activated. Since the comparison is finished, the reset must be activated, so again the NMOS must prevail, pulling down the node A.

Finally, there is another NMOS triggered by *end*, the pulse signal generated when the conversion is finished. This is necessary because the *EoC* signal needs more time to be generated, and it has to propagate through the NOR and the inverter before being able to pull down the node A. This generates a time window where another comparison is allowed, causing unnecessary power consumption. With this NMOS, the node A is pulled down before enough to stop this from occurring. The *end* signal is pulled down when *EoC* has had enough time to propagate and turn off the PMOS, keeping the node A to ground.

Figure 3.10 shows the structure of the delay inverter. The working principle is the same as that of the delay buffer (Fig. 2.8): a chain of buffers, where the chosen output propagates through the selected transmission gate to the node S. The difference appears at the last stages, where some modifications were applied to adapt it to the new loop architecture while reducing the delay of the circuit. For that same reason, also the sizes of the buffers

and the transmission gates are modified. The simulations considering parasitics show that the transmission gates struggle to pull up and down the node S because the associated capacitance is too big, increasing the delay and the power consumption due to cross conduction current in the NOR gate. For that reason, their size has been increased, in order to be able to drive the NOR gate with an acceptable delay. To drive this transmission gates, also the buffer size has been increased. In this way, the delay of the circuit, and thus the time the comparator is reset and the DAC has to settle, is reduced. This can be done thanks to the modifications implemented in the DAC, that have reduced its settling time from 120 ps to 60 ps. The comparator reset time is 40 ps, and it is always overcome thanks to the aforementioned improvements of the comparator reset loop.

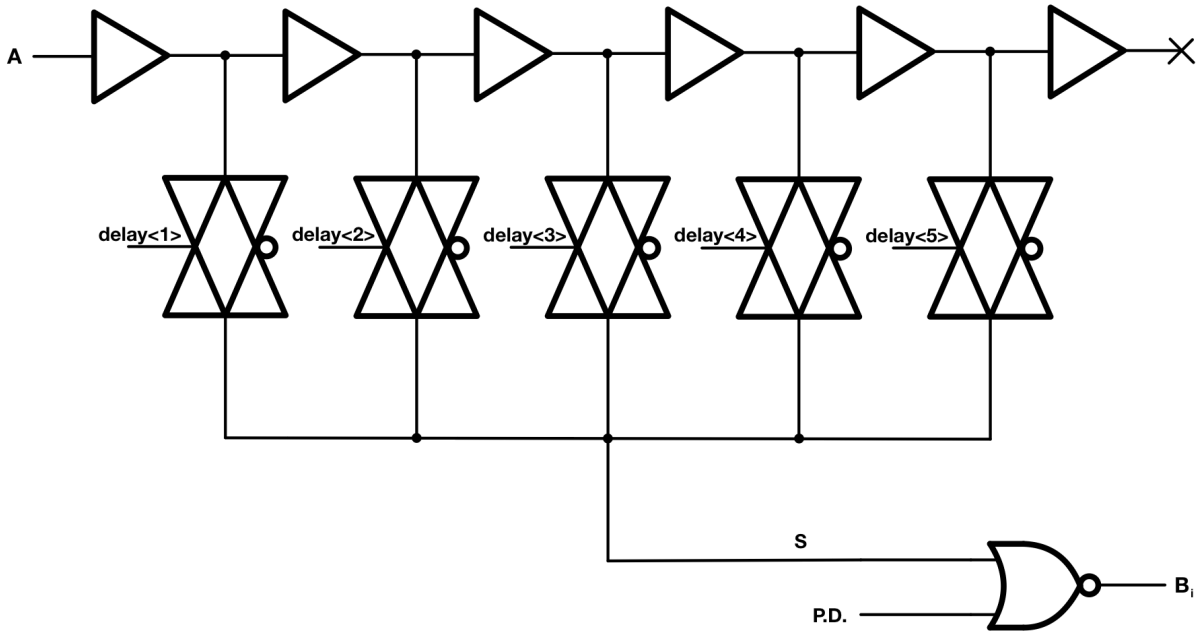

Figure 3.10: Structure of the delay inverter.

The structural differences between the delay inverter and the delay buffer appear at the last stages, between the output of the transmission gates (node S) and the output of the delay element (B for the delay buffer and B_i for the delay inverter). In the delay buffer, the voltage at B has to be the same as at the node S if *P.D.* is 0, while it has to be pulled down if *P.D.* is 1. In the delay inverter, if *P.D.* is 0, B_i is the complementary of the node S, and if *P.D.* is 1, B_i is pulled down, such that the NAND gate in the comparator reset loop (Fig. 3.9) is driven high, turning off the PMOS. Equations 3.1a and 3.1b represent the logic function of B and B_i.

$$\boldsymbol{B = S \cdot \overline{P.D.}} \tag{3.1a}$$

$$\boldsymbol{B_i = \overline{S} \cdot \overline{P.D.}} \tag{3.1b}$$

Applying the De Morgan laws, we get:

$$\boldsymbol{B = \overline{\overline{S} + P.D.}} \tag{3.2a}$$

$$\boldsymbol{B_i = \overline{S + P.D.}} \tag{3.2b}$$

As it can be derived from the equation, the only gate needed for the delay inverter is a NOR, while for the delay buffer also an inverter is required. The path from the node S to B_i has now just one stage, reducing the propagation delay.

The last difference between this reset control loop and the one presented in Chapter 3 is the removal of the external comparator reset $RST_{CMP\text{-}e}$, which activates the reset of the comparator when it is high and the temporal power down is active ($P.D.$=1). The reason for this change is that $P.D.$ already resets the comparator through the delay inverter. If $P.D.$ is 1, B_i is pulled down, the NAND gate that follows is driven high and the PMOS is turned off, so the node A cannot be pulled up. This means that until the temporal power down is pulled down, the comparator reset is active, making the addition of $RST_{CMP\text{-}e}$ useless. Also, the removal of the MUX that allows this behavior also benefits in terms of delay, power consumption and area.

3.3. New features

In the previous sections it is explained how the main goal of the project, reducing the delay of the logic to increase the maximum conversion rate, is achieved. However, other features need to be implemented in the logic, always under the condition of preserving the maximum conversion speed at the highest possible value. The features implemented in the proposed SAR logic are two: the comparator noise control and the metastability detector.

3.3.1. Comparator noise control

This subsystem allows the modification of the comparator's latch input referred noise during the conversion. In the first steps of the conversion, the priority in the comparison is speed, since the DAC settling takes more time. Instead, in the last steps the DAC settling is faster, so the comparison can take some extra time in exchange for reducing

the noise. Moreover, the first bits are redundant [3], thus more tolerant to noise, whereas the last bits are non redundant, hence requiring less noise.

To implement this behavior, the comparator has a controllable capacitance between the first stage (i.e., the dynamic preamplifier) and the second (i.e., the latch). Four capacitors are connected to the output of the preamplifier and controlled with four NMOS transistors which act as switches, connecting the capacitors to ground. The comparator noise control is responsible of activating the transistors, connecting and disconnecting the capacitors to the circuit.

All the four capacitors have the same size. To avoid different configurations that implement the same behavior, the circuit should have a hierarchy: a capacitor can be added to the circuit just if the previous one is also connected. However, this feature is not implemented in this subsystem, as it is the final user who selects the selection code, i.e., the capacitors that are activated.

The comparator noise control allows to set the input referred noise at the beginning and the end of the conversion, connecting the desired number of capacitors to the preamplifier output, and allows to select the step after which the noise changes. The circuit that implements this behavior is depicted in Fig. 3.11. The main signals are:

- **Initial selection code** (BW_{ini}<*1:4*>): this 4-bit signal contains the selection code for the desired input referred noise at the beginning of the conversion cycle.
- **Final selection code** (BW_{fin}<*1:4*>): as the previous one, but for the last steps of the conversion.
- **Step selection** (*sel*): this signal has 13 bits, one for each conversion step, and just one of them can be high. When the selected conversion step is completed, the comparator noise selection code changes from the initial to the final.

These three signals must be generated outside the SAR converter, since they are selected by the user. Besides, some more signals generated in other parts of the logic are used to obtain the desired behavior:

- **Comparison done** (*valid* and $\overline{valid}$): as in the comparator reset loop and the DAC control loop, the circuit is triggered when a comparison is completed, so when *valid* is pulled up and $\overline{valid}$ is pulled down.
- **General reset** (*Start*): while the change from the initial to the final selection code is triggered by *valid* and $\overline{valid}$, when the general reset is active the selection code must be the initial one, so the circuit is ready for the beginning of the next

conversion cycle.

- **Step signal** (*step<1:12>*): generated in the DAC control loop, each bit of this 12-bit signal is pulled up when the corresponding comparison step has been finished. The bit associated to the last comparison is not actually necessary, since no change is needed after the comparison is completed.

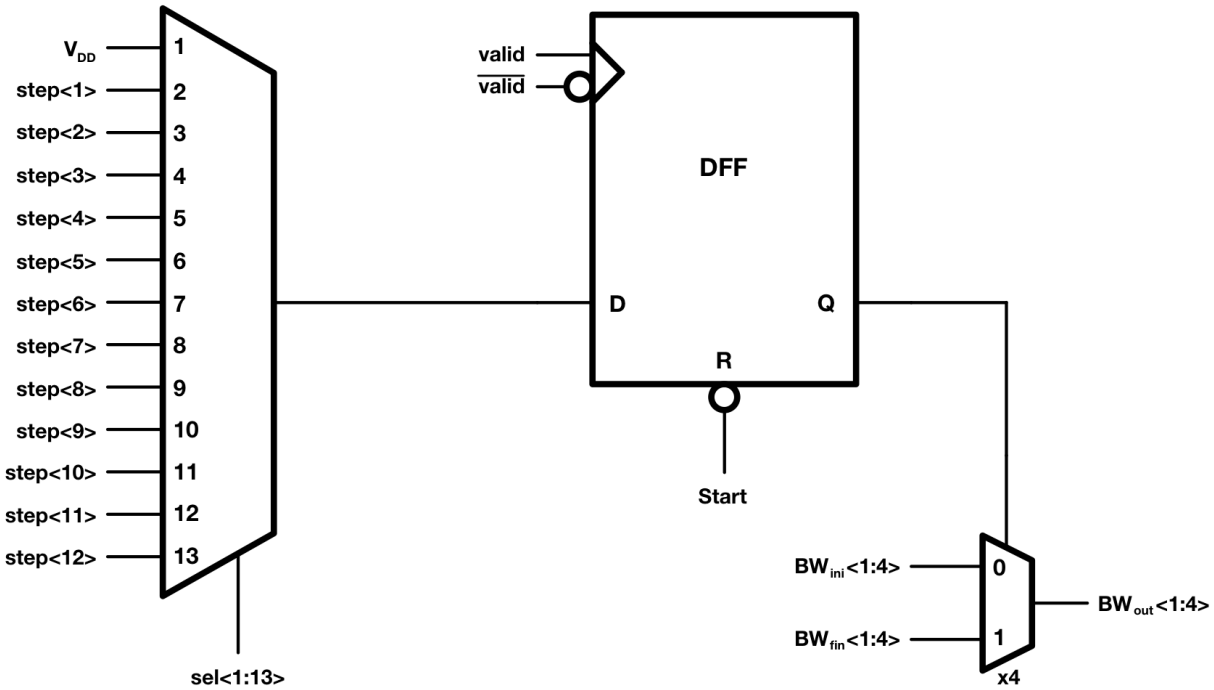

Figure 3.11: Architecture of the comparator noise control.

The step selection *sel* bit that is active allows the propagation of the corresponding input through the 13x1 MUX [6]. The goal of the MUX is that, when the selected comparison step is achieved and thus the DFF is triggered, the input of the DFF is 1, so the selection signal of the four 2x1 MUXs [6] is pulled up and the final selection code is propagated to the output. For that reason, the inputs of the 13x1 MUX are the step signals of the previous comparison. *step<i>* is always pulled up after *valid* and $\overline{valid}$ are pulled up and down, respectively, so it has almost one comparison cycle to propagate through the MUX.

If the selected step is the first one, the input of the DFF has to be pulled up before the first comparison is achieved. For that reason, the first input of the MUX is connected to V_{DD}, so the input of the DFF is 1 since the start of the conversion. On the other hand, if the selected step is the last one (*sel<13>*=1), the output of the DFF is never pulled up (since there are no more comparison steps). This case must be selected when the comparator noise control is not implemented, and the last input of the MUX can be

connected to ground rather than to *step<12>*.

Finally, to make sure that the initial selection code is selected, the comparator is reset during the sampling phase, using the *Start* signal.

With this configuration, the output of the circuit is modified at the beginning of the reset of the comparator, giving enough time to charge the capacitors to the reset voltage before the next comparison takes place.

3.3.2. Metastability detector

The last complication that may occur in the logic presented in Chapter 2 is the possibility of a metastable event: if the differential input voltage is close to 0 V, the comparator cannot make a decision in a limited amount of time. This results in a big delay in that conversion step, which means that the conversion is not achieved during the conversion period.

Even though this is not a problem of the digital logic but of the comparator, the solution can be implemented as a new block in the logic: the metastability detector. The goal of this circuit is to pull up a signal when a comparison takes enough time too be considered metastable. In the literature, two main architectures for the metastability detector stand out above the others: the programable timing window [10] and the NOR gate [11].

Programmable timing window metastability detector

In case of a metastable event, the conversion is not completed in the expected period. For that reason, this metastability detector gives a fixed time for every comparison to be achieved, pulling up a signal if that doesn't happen. For this behavior, the following inputs are needed:

- **Comparator reset** ($\overline{RST_{CMP}}$): when the comparator starts its operation (i.e., this signal is pulled up) the timing circuit has to be triggered, starting to count.
- **Comparison result** (CMP_P and CMP_N): if the comparison is achieved, one of this signals is pulled down, and the metastability detector output needs to remain low.
- **Time window control** (t_{CMP}*<1:5>*): even though the desired time window is always the same, PVT variations in the circuit make it necessary to trim the temporized delay, as it happens in the comparator reset loop. This signal allows choosing between five different delays to counteract these possible variations.

Figure 3.12 shows how the detector is implemented. At the beginning of each comparison,

the input of the delay inverter is pulled up. After a controllable delay, the output of the circuit is pulled down, triggering the low-edge TSPC flip-flop [6]. In a normal situation, when this happens the comparison has already been achieved and thus the XNOR gate is driven low. The FF stores this 0, and the output signal *MET* remains low. Instead, if there is a metastable situation, the comparison has not been achieved when the FF is triggered. The XNOR output remains high and *MET* is pulled up. Whether there is a metastable event or not, when the comparison step is finished and the comparator is reset, $\overline{RST_{CMP}}$ resets also the flip-flop, pulling down *MET*.

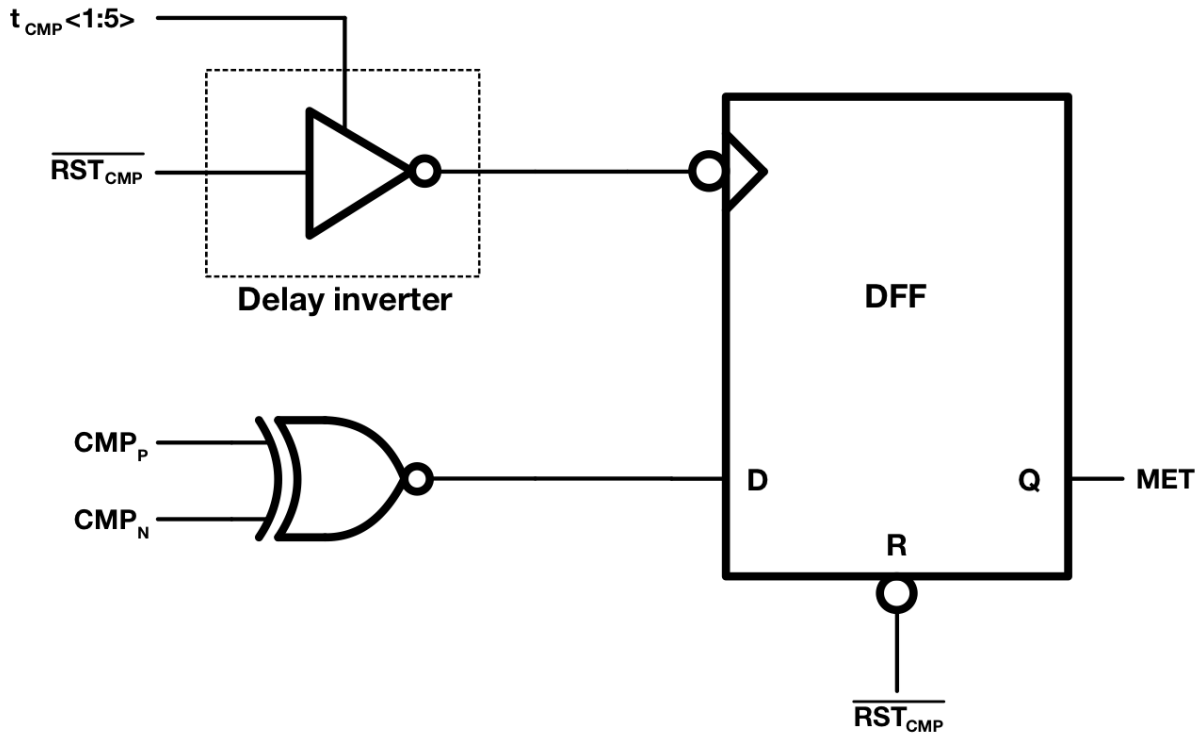

Figure 3.12: Architecture of the programmable timing window metastability detector.

The delay inverter used in this circuit is similar to the one in the comparator reset loop, showed in Fig. 3.10. The delay to consider a metastable event is set to 100 ps. This circuit has a reliable behavior, but it leads to a non negligible increase in terms of power consumption due to the delay inverter and the DFF.

NOR gate metastability detector

Another possible circuit able to detect the metastability is a NOR gate. The operation of this circuit is based on the idea that, when a comparison starts, the outputs of the comparator latch start discharging at different rates, until the voltage of one of them

crosses a threshold. In the case of a metastable event, both outputs discharge at almost the same rate. Then, a way to acknowledge a metastable event is to check if both comparator outputs are below a specific voltage. For that, a NOR gate designed to have an adequate threshold voltage can act as a metastability detector.

For this circuit, just the differential output of the comparator (CMP_P and CMP_N) is needed, as inputs of the NOR gate, and the only output is the metastability signal, *MET*. To obtain the desired threshold voltage, the aspect ratios of the PMOS and NMOS transistors has to be modified. It can also be helpful using low threshold voltage PMOS transistors and/or high threshold voltage NMOS. Due to the design of the comparator, the target NOR threshold is 750mV, 300mV above $V_{DD}/2$.

The circuit might seem an easier solution than the previous one, but as opposed to that, this solution is more sensible to PVT variations. A minor change in the gate threshold can mean an important variation in the time after a comparison event is considered a metastability. As a consequence, the circuit can take too much time before pulling up *MET*, not being able to finish the conversion in time, or might consider metastable a normal differential input, possibly giving a wrong digital value at the end of the conversion.

Application of the metastability detector to the logic

The two described architectures of a metastability detector inform that a metastable event is occurring. With this information, the logic needs to be modified in order to keep the circuit running and reach the next comparison step.

To start the comparator reset loop, the node A needs to be pulled up (Fig. 3.9). This can be done in two different ways:

- **Changing the threshold of the NAND gate**: as explained in the NOR gate metastability detector, if the threshold of the NAND gate is modified, a metastable event pulls down both inputs of the NAND, driving the gate high and turning on the NMOS that pulls down the node A, resetting the comparator. This solution doesn't add circuits to the logic, but it is sensible to PVT variations.
- **Adding an extra NMOS transistor**: an NMOS transistor between the node A and ground is included, in this case with its gate connected to *MET*. When a metastable event is detected, this transistor is turned on, and the node A is pulled down. This is a reliable solution, but it means some extra load at the node A that might reduce the speed of the loop.

For the DAC control loop, the *valid* signal needs to be pulled up, and so does one of the

latch inputs. Since the comparator outputs remain high during a metastable situation, a new signal must trigger the shift-register. This signal is pulled up either when the comparator takes a decision (i.e., *valid*=1) or when there is a metastable event (i.e., *MET*=1). This behavior can be achieved with an OR gate. To pull up one of the latch inputs, which in a normal situation are the complements of the comparator outputs, a NAND gate can be used. The inputs are connected to $\overline{MET}$ and CMP_P. In normal conditions, $\overline{MET}$ is high and the NAND gate acts as an inverter for CMP_P, while in a metastable situation, the gate is driven high. The output of this NAND gate can be used now as one of the inputs of the latch, while the other is still $\overline{CMP_N}$.

Unfortunately, adding all this circuitry has important drawbacks in terms of power consumption and delay. Considering the similar issues in the programmable timing window detector and the problems arising from PVT variations in the NOR gate detector, the metastability detector is still under investigation. However, malfunction associated with metastability can be avoided. The comparator outputs fall during a metastable event, reaching voltages below $V_{DD}/2$. Therefore, the NAND in the comparator reset loop is driven high, resetting the comparator. The DAC control loop can be also triggered, modifying the circuit that generates *valid* and $\overline{valid}$ so it uses a NAND gate instead of an XOR. A metastability event is able to pull *valid* up and $\overline{valid}$ down, starting the circuit operation. The corresponding latch is triggered, since the comparator outputs are low. The value stored in the latch is not necessarily the correct one, but the conversion is completed, and the error is of just 1 LSB. Thus, even though no signal is alerting of the metastable situation, the circuit operates normally and a conversion result is obtained.

3.4. Simulation results

Considering all the modifications applied to the circuit, an important reduction in terms of delay is expected, but in exchange of an increase in power consumption. In this section, all the variations in these two metrics are analysed, comparing the results with the ones obtained with the existing architecture both in each loop and considering the whole circuit.

To compare the results with the existing logic, the simulations have been run under the same conditions as the ones presented in Chapter 3: constant 50mV input voltage, nominal model of the transistors, 100º C, intermediate comparator reset time (*delay<3>*=1) and conversion frequency set to 250MHz. Figure 3.13 shows the main signals under these specifications. Some differences can be appreciated with respect to the results presented in Chapter 2: first, the first comparison takes the same time as the others (see $CMP_{P,N}$ in Fig. 3.13); second, *valid*, the comparator reset (CMP_{RST}) and the DAC control signals

($CTRL_{P,N}$<1:13>) are pulled up/down with a lower delay with respect to $CMP_{P,N}$; third, the DAC control signal $CTRLP_P$<2> is pulled down before *valid*, and fourth, the time the comparator reset is active has been reduced. This behavior matches the expected improvements from the modifications explained in this chapter.

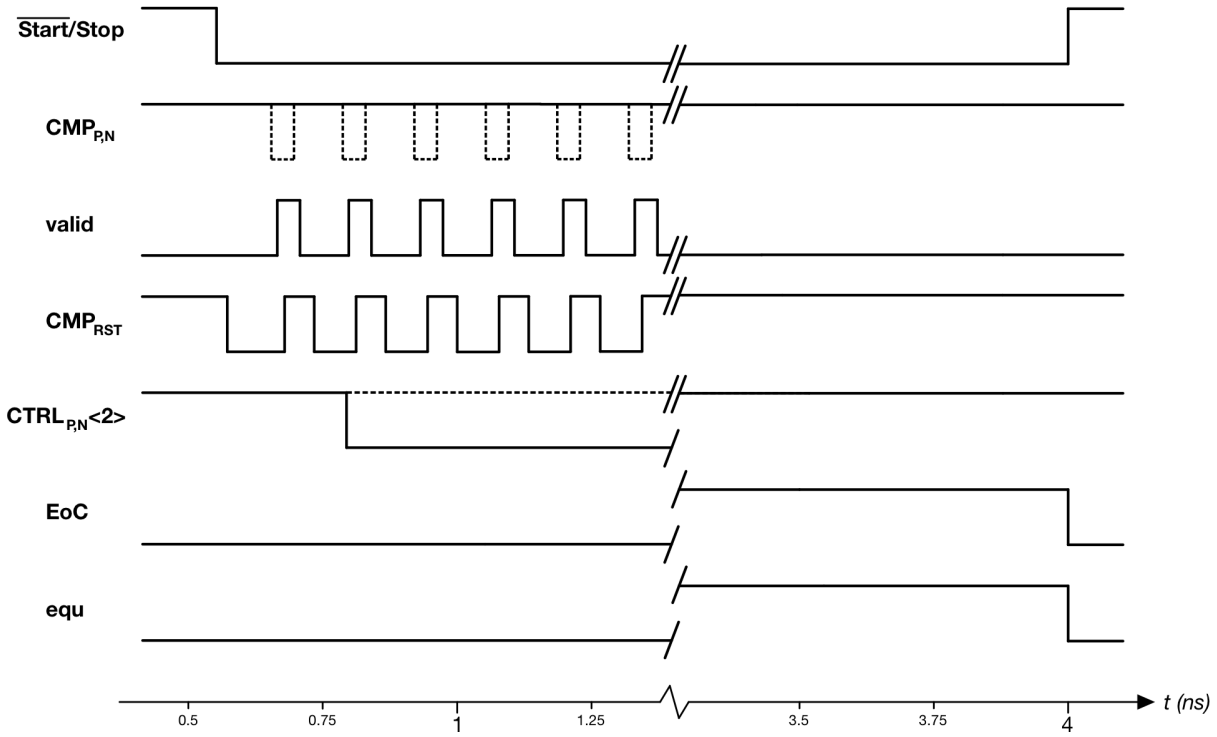

Figure 3.13: Waveform of the main signals of the proposed logic.

DAC control loop	
Logic delay	DAC settling
29.52 ps	72.63 ps
Comparator reset loop	
Logic delay	Comparator reset
34.81 ps	67.34 ps

Table 3.1: Delay on a loop step in the proposed logic.

Table 3.1 shows the different delays of a single step under these conditions. Comparing these results to the ones obtained for the existing logic (Table 2.2), it can be observed that the logic delay in both loops has been successfully reduced: almost a 50% in the DAC control loop and a 60% in the comparator reset loop. Also the comparator reset time and the time the DAC has to settle have been reduced (37% and a 46%, respectively), thanks

to modifications in the DAC that reduce its settling time, allowing to reduce the delay of the temporizer. The DAC settling time is still higher than the minimum, ensuring the correct behavior of the circuit.

Now, the delay of a single step, considering the 40 ps comparison time, is 142.15 ps, thus 1.848 ns for the 13 comparison steps. The total conversion time is then 2.112 ns, which is more than 1 ns below the target conversion period of 3.2 ns (i.e., target conversion frequency of 312.5 MHz), and almost 1.5 ns below the existing logic's conversion time of 3.472 ns. Thus, a major improvement can be observed in the proposed logic compared to the existing one, making it adequate to operate at the target conversion rate of 2.5 GHz inside the 8-channel time-interleaved architecture. Simulations performed in the logic without considering parasitics and using behavioral models for all the circuits but the digital logic show that **a single-channel SAR ADC can achieve a conversion rate of 375 MHz**.

As already mentioned, the circuit is sensible to PVT variations. For that reason, different corner simulations have been done, selecting the most adequate delay for each situation: *delay<4>* for the FF corner (fast PMOS, fast NMOS), *delay<2>* for the SS (slow PMOS, slow NMOS) and *delay<3>* for the FS (fast PMOS, slow NMOS) and SF (slow PMOS, fast NMOS). The results of these simulations are shown in Table 3.2. As it can be seen, similar results are obtained, always above the minimum DAC settling time of 60 ps and always improving the results obtained for the existing logic and achieving the target delay.

FF		**SS**	
DAC control loop			
Logic delay	DAC settling	Logic delay	DAC settling
23.82 ps	72 ps	35.32 ps	73.96 ps
Comparator reset loop			
Logic delay	Comparator reset	Logic delay	Comparator reset
33.21 ps	62.61 ps	38.42 ps	70.86 ps

FS		**SF**	
DAC control loop			
Logic delay	DAC settling	Logic delay	DAC settling
29.24 ps	71.44 ps	28.57 ps	71.36 ps
Comparator reset loop			
Logic delay	Comparator reset	Logic delay	Comparator reset
32.74 ps	67.94 ps	33.75 ps	66.18 ps

Table 3.2: Delay of a loop in the corner situations

In terms of maximum frequency, a simulation of a complete SAR ADC has been performed to analyse the logic behavior. Using verilog-A models for the other circuits in the converter (i.e., the DAC, the comparator and the sampler), the schematic model of the digital logic has been simulated for different conversion frequencies, both for the nominal transistor model and for the corners, with the goal of finding the maximum conversion frequency in each case. Simulations show that the maximum conversion rate is 375 MHz for all the models excepting the SS, which can't work faster than 346.25 MHz (still faster than the target 312.5 MHz). When a higher conversion frequency is applied to the ADC, the last bits are not generated, so the end of conversion generator is not started and the DFFs that display the digital output are not triggered. Thus, even if 12 comparisons are made, no result is obtained.

In terms of power consumption, the proposed logic has a worse behavior, due to both the modifications in the loops and the new subsystems. Considering just the digital logic, and with the same specifications applied to obtain the delay of each loop, the power consumption as function of the frequency is shown in Fig. 3.14 for nominal and corner models. The consumption in the nominal case is about 30% higher than in the existing logic and still follows a linear relation with frequency. The FF model shows a consistent increase in power consumption (25% higher than the nominal case), the SS one a more discreet decrease (10% lower than the nominal case) and the FS and SF ones show similar consumption to the nominal one, just slightly higher.

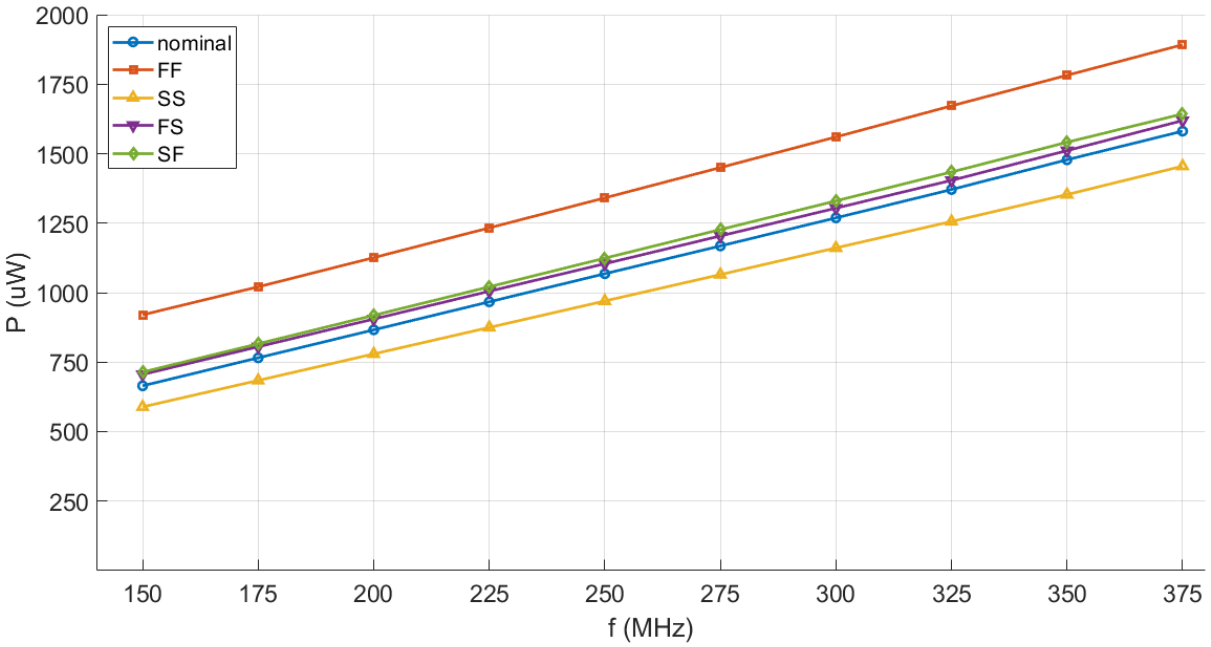

Figure 3.14: Power consumption for different frequencies: corners and nominal case.

Chapter 4

Conclusions

This thesis presents an improved digital logic for an existing SAR A-to-D converter that could not reach the target conversion frequency of 321.5 MHz. In the first chapter, the concept of analog to digital conversion is presented, highlighting the different kinds of converters and their principal figures of merit, to focus then on the SAR ADC and finally describe the specific implementation that has been analysed and improved in this work.

Chapter 2 explains in depth the behavior of the existing SAR logic, analysing all the implemented features and remarking the most important elements of the system and their main limitations. It has been proved that the two main loops of the circuit, the comparator reset loop and the DAC control loop, are spending an important part of the time assigned to the conversion, and that the time allocated for the DAC settling has been overestimated. Besides this, a lack of robustness against metastable situations has been observed.

After the extensive analysis of the existing logic, the modifications implemented in the logic to cope with the observed problems are shown in Chapter 3. A smart implementation of the DAC control loop allows to significantly reduce the delay of this subsystem, whereas the comparator reset loop has been completely redesigned to achieve remarkable improvements as well. The temporizer has also been modified to adapt to the new DAC and its reduced settling time. A similar circuit with increased size can be used, with a negligible penalty in terms of area. The metastability issue has been also solved with minor modifications in the circuit that also allow to slightly reduce the power consumption. Finally, an extra feature has been implemented in the circuit to improve the operation of the comparator. The proposed logic has been simulated and compared to the existing one, showing major improvements in terms of conversion rate that outperforms the initially proposed goals.

The proposed logic displays a robust performance and achieves major improvements in

the most important elements of the circuit, keeping the advantages of the existing logic while eliminating the limiting factors. The result is a fast, reliable and precise SAR ADC that can compete with the most advanced converters at medium and high frequencies thanks to its time-interleaved architecture. The objectives set for this thesis have been widely overcome, ensuring that the converter will feature the desired behavior also after fabrication.

4.1. Future lines

The results obtained in this work not only fully comply with the project specifications, but also open the possibility to a variety of challenging lines of work.

At research level, the increase of the maximum conversion speed is due to modifications in the digital logic of the DAC, but the comparator delay represents now a 28% of a bit evaluation time. Therefore, a new version of the comparator with reduced decision delay would mean another significant improvement in terms of frequency. In the same line, a new DAC topology with reduced settling time would also improve the converter speed.

Regarding the logic, the most important elements have been optimized, but there is always research from which all levels of the logic can benefit: modified architectures, new structures for specific elements and technological development. Therefore, the last advancements in these fields must be always under special attention to extract every possibility of improvement for the logic.

At application level, the proposed logic is adapted to a specific project that requires 13 comparisons to obtain 11 redundant bits at a maximum frequency of 312.5 MHz. However, the benefits of the logic are not limited to this specific case, but a variety of specifications can be met if the trade-off between resolution and maximum operating frequency is respected. Thus, both high frequency-low resolution and low frequency-high resolution applications can benefit from an adaptation of this converter.

Bibliography

[1] B. Murmann, "The successive approximation register adc: a versatile building block for ultra-low-power to ultra-high-speed applications," IEEE Communications Magazine, vol. 54, no. 4, pp. 78–83, 2016.

[2] P. Harpe, "Successive approximation analog-to-digital converters: Improving power efficiency and conversion speed," IEEE Solid-State Circuits Magazine, vol. 8, no. 4, pp. 64–73, 2016.

[3] F. Kuttner, "A 1.2 v 10b 20msample/s non-binary successive approximation adc in 0.13/spl mu/m cmos," in 2002 IEEE International Solid-State Circuits Conference. Digest of Technical Papers (Cat. No. 02CH37315), vol. 1, pp. 176–177, IEEE, 2002.

[4] R. Schreier, G. C. Temes, et al., Understanding delta-sigma data converters, vol. 74. IEEE press Piscataway, NJ, 2005.

[5] S. Brenna, A. Bonfanti, and A. L. Lacaita, "A 70.7-db sndr 100-ks/s 14-b sar adc with attenuation capacitance calibration in 0.35-μm cmos," Analog Integrated Circuits and Signal Processing, vol. 89, no. 2, pp. 357–371, 2016.

[6] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, Digital integrated circuits, vol. 2. Prentice hall Englewood Cliffs, 2002.

[7] Y. Cao, Y. Chen, Z. Ni, F. Ye, and J. Ren, "An 11b 80ms/s sar adc with speed-enhanced sar logic and high-linearity cdac," in 2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 18–21, IEEE, 2018.

[8] J.-S. Park, J.-M. Jeon, J.-H. Boo, J.-H. Lee, K.-I. Cho, H.-J. Kim, G.-C. Ahn, and S.-H. Lee, "A 2.2 mw 12-bit 200ms/s 28nm cmos pipelined sar adc with dynamic register-based high-speed sar logic," in 2020 IEEE Asian Solid-State Circuits Conference (A-SSCC), pp. 1–2, IEEE, 2020.

[9] Q. Yu, X. Zhou, K. Hu, Z. Huang, H. Chen, X. Si, J. Yang, and Q. Li, "A 9.08 enob 10b 400ms/s subranging sar adc with subsetted cdac and pdas in 40nm cmos," in

ESSCIRC 2021-IEEE 47th European Solid State Circuits Conference (ESSCIRC), pp. 391–394, IEEE, 2021.

[10] A. Shikata, R. Sekimoto, T. Kuroda, and H. Ishikuro, "A 0.5 v 1.1 ms/sec 6.3 fj/conversion-step sar-adc with tri-level comparator in 40 nm cmos," IEEE Journal of Solid-State Circuits, vol. 47, no. 4, pp. 1022–1030, 2012.

[11] P. J. Harpe, C. Zhou, Y. Bi, N. P. Van der Meijs, X. Wang, K. Philips, G. Dolmans, and H. De Groot, "A 26 μw 8 bit 10 ms/s asynchronous sar adc for low energy radios," IEEE Journal of Solid-State Circuits, vol. 46, no. 7, pp. 1585–1595, 2011.

List of Figures

List of Tables