AGH University of Science and Technology
Faculty of Physics and Applied Computer Science

Master thesis

Piotr Rymaszewski

major: **technical physics**

specialisation: **solid state physics**

Development of 12-bit ADC for particle physics applications using deep-submicron CMOS technology

Supervisor: **dr hab. inż. Marek Idzik**

Cracow, August 2013

Aware of criminal liability for making untrue statements I declare that the following thesis was written personally by myself and that I did not use any sources but the ones mentioned in the dissertation itself.

.....
(Signature)

Cracow, August 2013

The subject of the master thesis and the internship by Piotr Rymaszewski, student of 5th year major in technical physics, specialisation in solid state physics

The subject of the master thesis: **Development of 12-bit ADC for particle physics applications using deep-submicron CMOS technology**

Supervisor:	Dr hab. inż. Marek Idzik
Reviewer:	Dr inż. Krzysztof Świentek
A place of the internship:	WFiIS AGH, Kraków

Programme of the master thesis and the internship

1. Discussion with the supervisor on realization of the thesis.
2. Collecting and studying the references relevant to the thesis topic.
3. The internship:
   - Design and simulations of integrated analog–to–digital converter.
   - Design of physical layout of circuit.
4. Ordering and first analysis of the obtained results.
5. Final analysis of the results obtained, conclusions – discussion with and final approval by the thesis supervisor.
6. Typesetting the thesis.

Dean's office delivery deadline: August 2013

....

(signature of Head of Department) (Supervisor's signature)

Supervisor's review

Author wishes to express sincere gratitude to his supervisor dr hab. inż. Marek Idzik who provided invaluable support and guidance throughout the research.
Special thanks are also due to dr Jan Kapłon, Dominik Przyborowki, Jakub Moroń and Michał Dwużnik whose knowledge and help were vital to the author.

Table of contents

Introduction

Current times are very exciting for High Energy Physics (HEP) – results from experiments carried out at Large Hadron Collider (LHC), the biggest particle collider up to date, state that a particle very similar to Higgs boson have been found. This is a discovery of a great magnitude, since it delivers long sought after informations about particle interactions and further confirms that Standard Model, theory currently used to describe particles and their behaviour, is correct. In light of this discovery future HEP experiments are being planned to determine, with precision greater than achievable by LHC, exact nature and properties of this newly found particle. To reach these goals new colliders and detectors will require electronics capable of delivering more precise data than electronics used in current experiments.

Presented thesis is focused on read-out microelectronics that might be used in future HEP experiments. The goal of this work is design of 12-bit analog-to-digital converter with very low power consumption (in range of single mW) capable to work with 40MHz sampling clock.

The thesis is divided into four chapters. First contains brief overview of High Energy Physics experiments – starting with physics behind them, followed by review of LHC, largest currently running experiment, and a role of electronics in its operations. Here also motivations for this works are presented.

Chapter two reviews basic definitions and parameters (both dynamic and static) connected with ADCs in general, which will be used throughout the thesis. Furthermore most popular ADC architectures are presented and example applications are mentioned.

In third chapter various approaches to successive approximation ADC are reviewed in detail accompanied by results of Matlab simulations of power consumption of each configuration. Based on presented informations the choice of configuration used in this design is justified. In this chapter also general considerations about SAR ADC building blocks are presented.

Fourth chapter presents in detail designed converter – chosen architecture for each block and it's working principles are explained (heavily relying on theories and equations from previous chapter). Here also are presented simulations results of each block separately and ADC as a whole.

Thesis is than concluded with a summary which reviews achieved goals and a brief discussion of potential future work.

1 | High Energy Physics experiments

1.1 Physics in HEP experiments

The main theory currently used to describe particle physics is Standard Model. It was developed in 1970s by Steven Weinberg, Sheldon Glashow and Abdus Salam and is still applied to explain results of many high energy physics experiments. According to this theory the Universe is built out of structureless particles that can be divided into two groups having 6 members each:

- leptons (having a unity electric charge e^-): e^-, ν_e, μ^-, ν_μ, τ^-, ν_τ
- quarks (having fractional electric charge $\frac{2}{3}e^-$ or $-\frac{1}{3}e^-$): up, down, top, bottom, strange, charm

Interactions in Standard Model are described as exchange of bosons, different for each interaction (photons for electroweak, gluons for strong, W^+, W^-, Z^0 for weak and not verified experimentally gravitons for gravitational interactions).

In the years following it's introduction studies of particle physics using Standard Model shown that for energies in range of teraelectronovolts electromagnetic interaction and weak interaction can be in fact described by one mechanism (called electroweak interaction) - this lead to extrapolation of theory, saying that in fact for high enough energy scale all interaction become one, as presented at Figure 1.1. To better describe unification of electromagnetic and weak interactions supersymetric particles were postulated, which were supposed to enable the unification mechanism.

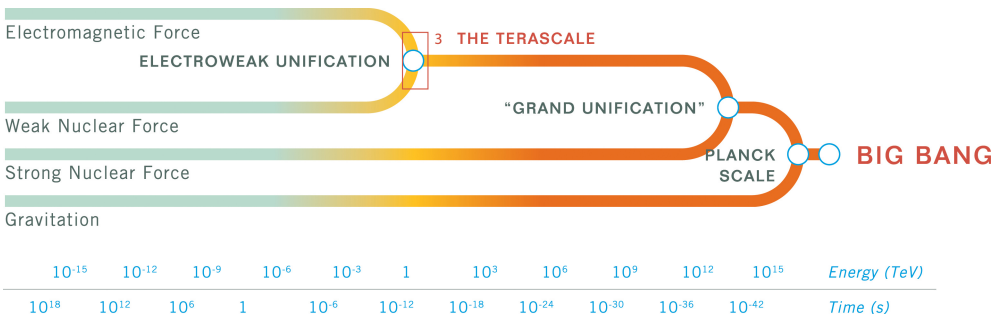


Figure 1.1: Energy scale for unification of fundamental interactions. [1]

Originally Standard Model postulated that all bosons are massless, but experiments UA1 and UA2 carried out in CERN in 1980s proved otherwise. This discovery lead to modification of Standard Model - Higgs mechanism was introduced, explaining inconsistency between theory and experimental data by adding new particle, Higgs boson, which by interaction with other particles gave them mass.

To experimentally verify existence of supersymetric particles and Higgs boson Large Hadron Collider (LHC) was build at CERN - first HEP experiment capable of producing and observing particle interactions at energies in teraelectronovolt range. At 4^{th} July 2012 first results of this experiment were published showing that a particle consistent with characteristics of Higgs boson was observed [2] (Figure 1.2). Despite of this tremendous achievement experiments at LHC are not finished – though new particle was found it's exact characteristics must be further examined to verify if it is indeed a Standard Model Higgs boson or other kind of boson as predicted by theories which go beyond Standard Model [3].

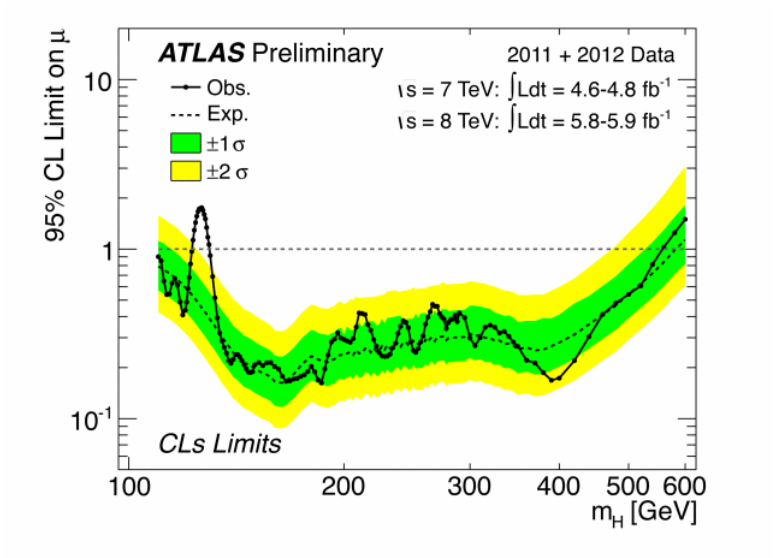


Figure 1.2: Experimental limits from ATLAS on Standard Model Higgs production in the mass range 110-600 GeV. The solid curve reflects the observed experimental limits for the production of a Higgs of each possible mass value (horizontal axis). The region for which the solid curve dips below the horizontal line at the value of 1 is excluded with a 95% confidence level (CL). [2]

Despite this newest discoveries one must remember though, that Standard Model is by no means the final model of particle physics - it presents a very good description of particles as we currently understand them, but it also has its weaknesses:

- it does not incorporate gravitational interactions,
- it requires many additional parameters to be introduced to explain some phenomena (e.g. neutrino oscillations, two independent masses for weak bosons),
- it does not explain some of particle's quantum numbers (e.g. electric charge, colour).

1.2 Present and future HEP experiments

As have been mentioned in previous section currently the largest experiment of high energy physics is Large Hadron Collider at CERN. It is situated in 27 km long tunnel, formerly used by Large Electron Positron Collider. LHC is designed to collide two proton beams with maximum energy of 8 TeV in the centre-of-mass or two lead ions beams with energy of 5.5 TeV per nucleon pair. Acceleration of beam to maximal energy is a complicated process consisting of four stages (in case of proton beams) [4]:

- Linac-2 – production of proton beam with energy of 50 MeV
- PSB – acceleration of beam from Linac-2 to kinetic energy of 1.4 GeV
- PS – further acceleration of beams up to 26 GeV
- SPS – final injector for LHC (also providing beam for COMPASS and CNGS projects), achieving beam energy of 450 GeV

After injection into LHC beam is further accelerated to maximal energy and kept on right track using superconducting magnets. Two proton beams circulate LHC ring in opposite direction inside two separate channels and are intersected only in four places where main experiments are located (schematic view of LHC complex in presented in Figure 1.3):

- ATLAS – largest of LHC experiments, general purpose detector (search for Higgs boson, supersymetric particles, dark matter, etc.)
- CMS – second largest experiment, also general purpose
- LHCb – designed to study asymmetries between B and $\bar{B}$ mesons
- ALICE – build to study properties of strongly interacting matter at extreme energy densities (quark-gluon plasma) during lead ions collisions

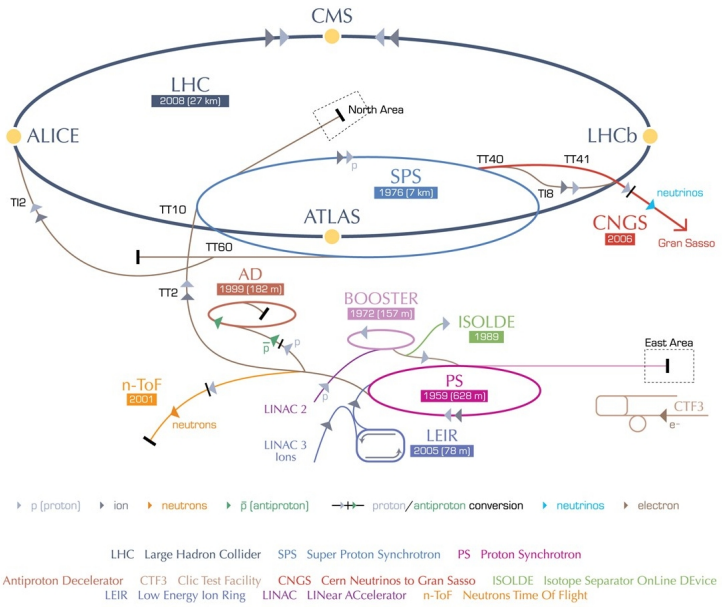


Figure 1.3: Schematic of LHC complex with indiaction of all experiments and booster rings. Reproduced from [5].

It was foreseen that some of the innermost parts of detectors will suffer from performance degradation due to radiation effects after a few years of running the experiment. To remedy this a long-term plan to remove damaged parts and replace them with upgraded substitutes was adopted. This plan assumes a series of upgrades, done in two phases, to end approximately at year 2021, leading to increase in collision energy of proton beams to 14 GeV and ten-fold increase in luminosity (number of events per second) – hence the name of final configuration of the machine is High Luminosity LHC. This will allow to further improve on accuracy of studies (e.g. measure more precisely mass of newly discovered boson) and also extend possibilities in terms of new particles and phenomena discoveries.

Although increase in energy and luminosity in HL LHC will lead to possibility of more precise measurements, there is a fundamental barrier that limits accuracy – both LHC and HL LHC use proton beam. Because protons have their own internal structure (two up quarks and one down quark) their collisions produces high background, which prevents achieving high precision needed to answer questions about Higgs boson mass, spin, parity, etc. A solution is to use structureless particles – leptons, which collisions should be much cleaner and therefore observations and measurements of new particles would be made easier. Since electrons are only stable leptons, they are natural choice in this case, but their low mass (about 2000 times lower than that of a proton) causes them to radiate their energy much more rapidly when their path is curved (bremsstrahlung radiation is proportional to m^{-4}). Therefore successors to LHC will be linear colliders and plans together with research & development work for two possible candidates (ILC – International Linear Collider and CLIC – Compact Linear Collider) have been undergoing for past few years – their design specifications can be found in Table 1.1.

Table 1.1: Basic design parameter for the ILC and the CLIC accelerators [6].

Parameter	ILC	CLIC_{500GeV}	CLIC_{3TeV}
centre-of-mass energy $[GeV]$	500	500	3000
peak luminosity $\left[\frac{1}{s \cdot cm^2}\right]$	$2 \cdot 10^{34}$	$2.3 \cdot 10^{34}$	$5.9 \cdot 10^{34}$
pulse rate $[Hz]$	5	50	50
number of bunches per pulse	∼3000	354	312
bunch spacing $[ns]$	330	0.5	0.5
particles per bunch	$2 \cdot 10^{10}$	$6.8 \cdot 10^{9}$	$3.7 \cdot 10^{9}$
accelerating gradient $\left[\frac{MV}{m}\right]$	31.5	80	100
energy loss due to bremsstrahlung $\left[\frac{\Delta E}{E}\right]$	0.03	0.07	0.28
total AC power consumption $[MW]$	230	271	582

General layouts of both accelerators are similar (therefore only ILC's schematic is presented – Figure 1.4) but their build methodology, final specifications and experimental conditions are very different. Full comparison of the two is far out of scope of this work but two differences will be pointed out:

- centre-of-mass energy – while ILC uses 1.3GHz Superconducting RF cavities to accelerate electrons over a course of over 30 km to energy of 500GeV, CLIC is considered to be designed in multi-stage way to achieve energy of 500GeV, 1.5TeV or 3TeV depending on stage. Operation frequency of CLIC is 30GHz, which leads to device length of 37.5km for maximal energy,
- bunch spacing - much shorter time between subsequent bunches (0.5ns for CLIC vs. 330ns for ILC) makes a very significant difference for electronics needed for detectors - time-tagging combined with good pileup management are essential for CLIC, while in ILC each bunch can be processed separately.

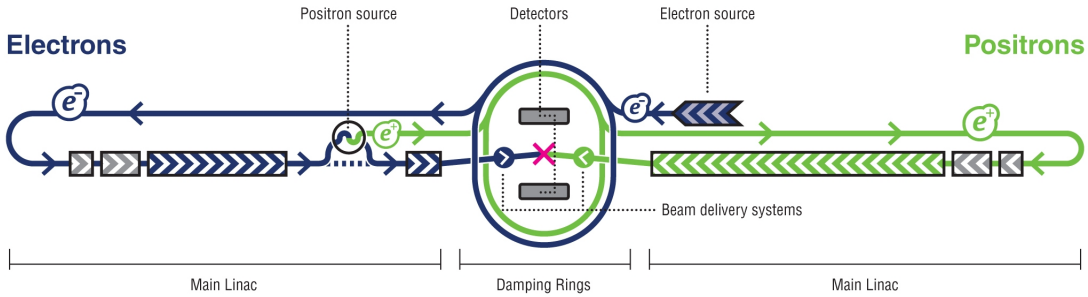


Figure 1.4: Schematic of LHC complex with indiaction of all experiments and booster rings [1].

Choice between two designs will be determined be results from LHC, so in light of recent discovery of new boson ILC seems like a more probable candidate. On the other hand data analysis from ATLAS and CMS suggest that there is no new physics below 1TeV [7], so if LHC will find some signs of supersimetric particles in higher energies proceeding with CLIC will be the only choice.

1.3 Read-out electronics

Detectors used in each of experiments described in previous section are very different from one another - they are designed with different specifications (detecting different kinds of particles, working with different collision energies and luminosities, etc.) but all of them require very specific read-out electronics and its design have to take into consideration not only appropriate functionality but also many additional factors:

- effects connected to prolonge exposure to high levels of radiation (increased leakage current, shifts in transistor's threshold voltages, signle event upsets, etc.)
- very high number of read-out channels combined with small avaiable space

– need to minimize amount of used materials (to reduce secondary interactions with beam collision products)
– high relaiability (due to as compact construction of detectors as possible and presence of high energy particle beams, all repairs and replacements of parts are problematic)

All those factors make it necessary to design read-out electronics as Application Specific Integrated Circuits (ASIC) instead of using complex systems of discrete elements. As example of such ASIC system a read-out chain for LumiCal detector (part of detector for ILC) is presented on Figure 1.5.

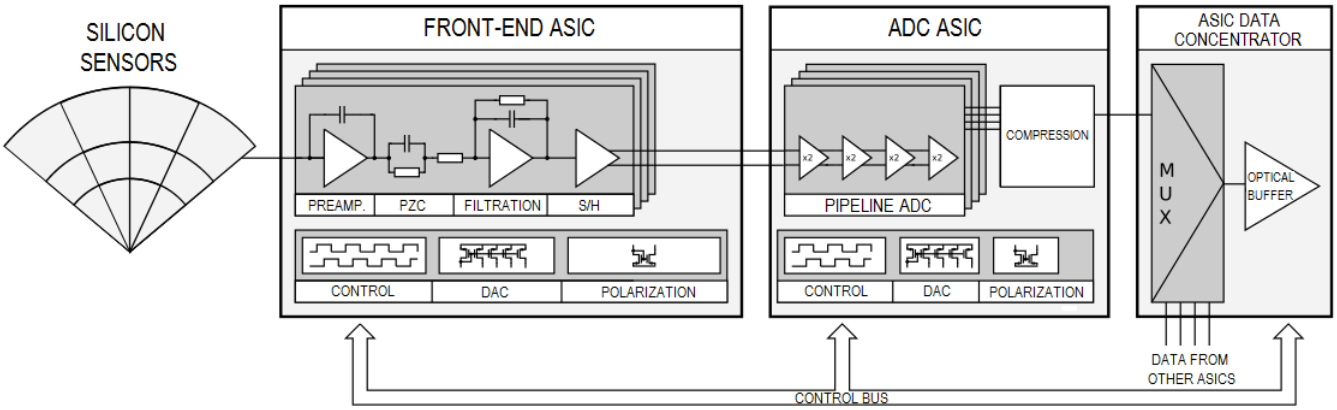


Figure 1.5: LumiCal's readout electronics flow chart. Reproduced from [10].

LumiCal is a sandwich calorimeter (it is build out of staggered layers of tungsten absorber and silicon sensors). When an electron or positron pass through such structure they deposit an electric charge in sensor layer and pass through it, but when they encounter tungsten they quickly lose their momentum resulting in bremsstrahlung radiation. This causes (in environment of heavy tungsten nuclei) creation of new electron - positron pairs – so called electromagnetic shower occurs. Shape of this shower and number of penetrated layers is determined mainly by kind of particle which passes through and its energy. Role of read-out electronics is to measure the charge collected by each sensor and pass it to Data Acquisition system (DAQ). Front-end ASIC is supposed to extract information from silicon sensor, shape and amplify it and store the information in sample & hold device. Because of ease of transmition and data processing signal from front-end is converted to digital form using an analog-to-digital converter. Output stage – data concentrator – passes informations from ADC to DAQ via optical buffer.

1.4 Motivations for this work

As was mentioned in previous sections future HEP experiments will require more precise measurement electronics than used presently. This thesis is a research & development work investigating if ADC with parameters of potential interest to future experiments (low power,

12-bit resolution with 40MHz sampling frequency, 144 μm pitch) is feasible to design using 130nm technology. Inspiration for this attempt is very high activity related to SAR ADC resulting in quite rapid development of new variations and improvements for this architecture – this trend can be seen in number of publications on this topic in recent years (as presented in Figure 1.6). Focus will be especially given to new method of capacitive DAC switching which lead to very significant reduction in power consumption.

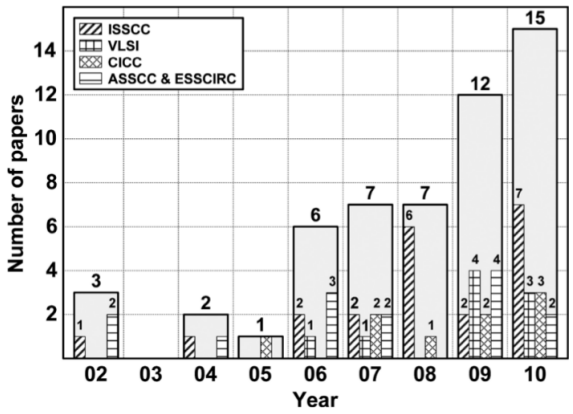


Figure 1.6: Number of articles about SAR ADC published in recent years. [8]

2 | Overview of analog-to-digital converters

2.1 Basic definitions

An analog-to-digital converter is a device connecting analog and digital signal domains - it translates an analog signal (continuous in time and amplitude) into a digital word (composed of few signals with binary quantised amplitudes denoted 0 for $V_{ref,min}$ and 1 for $V_{ref,max}$ – see figure 2.1).

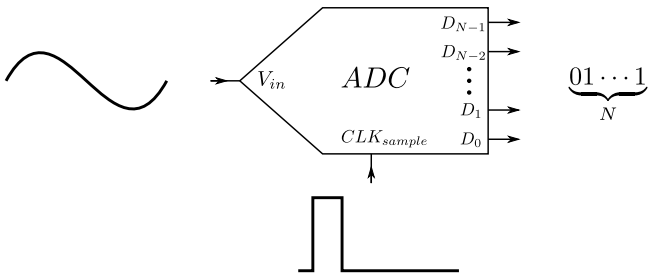


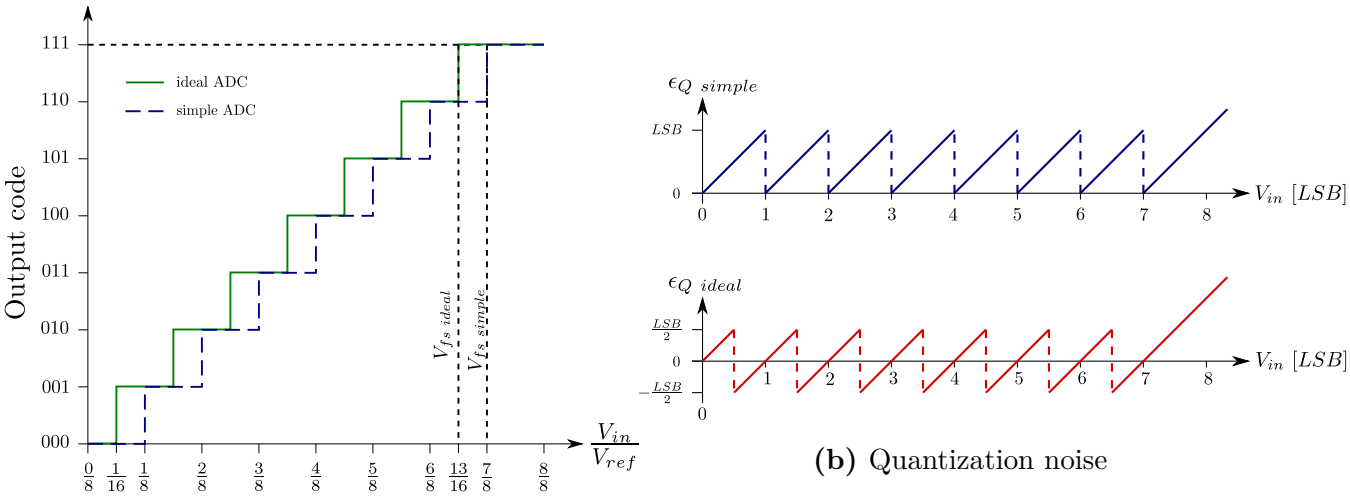
Figure 2.1: Schematic representation of operation of ADC.

Due to the very nature of this process translation will never be ideal – for two input samples ADC's output will be different only if the samples differ by more than minimal voltage recognizable by ADC. This minimal value is called *Least Significant Bit (LSB)* and for N-bit converter is defined as [9]:

$$LSB \equiv \frac{V_{ref}}{2^N} \tag{2.1}$$

This inaccuracy results in multi-step input-output characteristic (example shown in Figure 2.2a) and introduces so called quantization noise – difference ϵ_Q between real value of input and its quantisized substitute (presented in Figure 2.2b). Figure 2.2 contains plots for two examples of ADC – "simple ADC" which functions exactly as described in previous paragraph (change of input voltage by 1 LSB results in change of output code by one) but, as can be observed by comparing equations 2.19 and 2.20, exhibits lower signal-to-noise ratio than "ideal ADC" (same characteristics as simple one but with added offset $V_{offset} = \frac{1}{2}LSB$). This claim

is proven in section 2.1.2.



(a) Inpout-output characteristics

(b) Quantization noise

Figure 2.2: Consequences of input signal quantization shown for 3-bit ADC.

Output of ADC can be therefore expressed as (assuming binary-weighted output bits):

$$V_{out} = V_{ref} \cdot \sum_{i=0}^{N-1} D_i 2^i \tag{2.2}$$

where D_i represents value of i-th digital output.

On figure 2.2a two characteristic voltage levels can be noticed: V_{ref} – reference voltage and V_{fs} – full scale voltage. They are connected through relation:

$$V_{fs,simple} = V_{ref} - 1 \cdot LSB \tag{2.3}$$

$$V_{fs,ideal} = V_{ref} - \frac{3}{2} \cdot LSB \tag{2.4}$$

Combining equations 2.1 and 2.3 leads to other, also commonly used, definition of LSB:

$$LSB = \frac{V_{fs}}{2^N - 1} \tag{2.5}$$

2.1.1 Static parameters

There are many characteristics that can be used to measure how much given ADC differs from an ideal one. In this section those used when sampling static or slowly changing signals will be introduced.

Offset error

As mentioned in previous section an ideal ADC has an offset of $\frac{1}{2}LSB$ – any deviation from this value is considered an offset error (also known as zero-scale error). To put this in other words – difference between $\frac{1}{2}LSB$ and voltage causing first ADC transition is an offset voltage. Origins of this error can be different for different architectures (offset in comparators in flash ADC, offset in DAC for SAR, etc.) but in general it is correlated with mismatch of components of an ADC or changes in reference voltages.

Gain error

Gain error, also called slope factor error, is a difference in slope of straight line drawn through the transfer characteristics and the slope of corresponding line of an ideal ADC.

Full scale error

Full scale error is in principle very similar to offset error – it is difference between ideal value of full-scale voltage ($V_{fs} = V_{ref} - \frac{3}{2}LSB$) and measured one that triggers transition to last output code available. This error is a result of both offset error and gain error.

Differential NonLinearity (DNL)

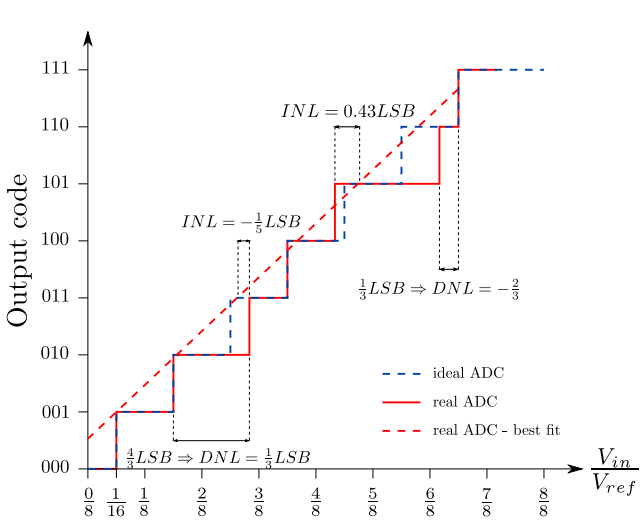
While three kinds of errors mentioned above are not very severe since they can be removed with measurement calibration, DNL and described next INL are more important.

For an ideal ADC difference in input voltage for which output code change $\Delta V_{in\ change}$ is, by definition, equal to 1 LSB. For real ADC value of $\Delta V_{in\ change}$ will most likely differ for each output code due to elements mismatch, process variation, etc. Differential nonlinearity is a measure of how much $\Delta V_{in\ change}$ changes from code to code and can be defined as [11, 12] (example of transfer curve of an ADC exhibiting DNL errors is presented in Figure 2.3a):

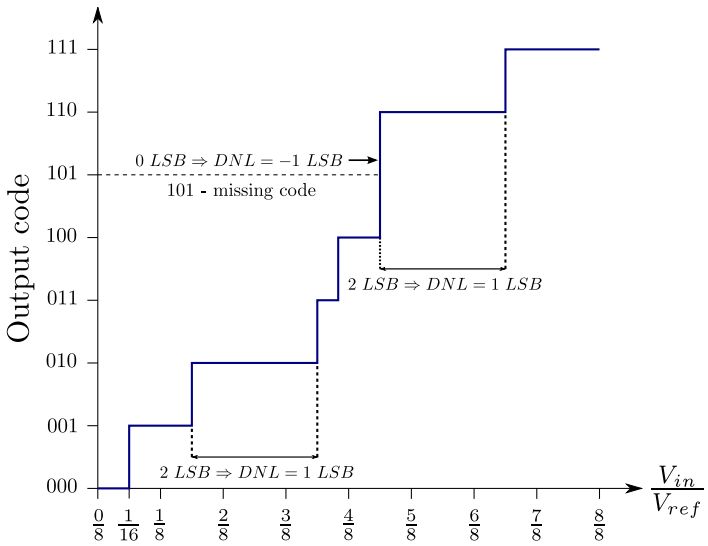
$$DNL(m) = \frac{\Delta V_{in\ change}(m) - LSB}{LSB} = \frac{V_{in}(m) - V_{in}(m-1) - LSB}{LSB}$$
$$[DNL(m)] = LSB \tag{2.6}$$

Value of DNL provides also information about missing codes [12] (ass can be observed in Figure 2.3b):

- $DNL(m) \leq -1\ LSB \Rightarrow m$-th code will be missing
- $DNL(m) \geq 1\ LSB \Rightarrow m$-th code is present, $(m+1)$-th presence depends upon $(m+2)$-th transition value



(a) Characteristics with INL and DNL errors.



(b) DNL as indication of missing codes.

Figure 2.3: Example of INL and DNL errors for 3-bit ADC.

Integral nonlinearity (INL)

Information about converters linearity is obtained by calculating INL [11, 12]:

$$INL(m) = \frac{V_{meas}(m) - V_{fit}(m)}{LSB} \qquad [INL(m)] = LSB \tag{2.7}$$

where $V_{meas}(m)$ is measured value of m-th transition step and $V_{fit}(m)$ is value of this level calculated from straight line fit to transfer function (as presented in Figure 2.3a). In literature two ways of fitting are reported:

- fit only through first and last point of transfer characteristics
- use best fit to fit all point from characteristics (used in this work)

Integral non-linearity measures monotonicity of converter – if highest value of INL is below $\frac{1}{2}LSB$ converter is monotonic [11]. One of the most important characteristics of an ADC is its *Effective Number Of Bits (ENOB)*, which informs about realistic resolution of converter during normal operation. It can be calculated using INL as [13]:

$$ENOB = \log_2\left(\frac{2^N}{\sqrt{1 + \frac{12}{2^N - 2} \cdot \sum_{m=1}^{2^N-2} INL(m)^2}}\right) \tag{2.8}$$

2.1.2 Dynamic parameters

Dynamic parameters inform about converter's behaviour when sampling fast changing signals. This shows influence of noise, sampling time uncertainty, nonlinear distortions, etc. Results obtained through dynamic parameters depend not only on ADC itself, but also on signals used as input (amplitude, frequency) and sampling clock (frequency, jitter). This makes dynamic parameters measurement more demanding than static ones.

One of commonly used method of measurement is through analysis of discrete Fourier transform (DFT) of converters response to input sinus signal with amplitude $\frac{V_{ref}}{2}$. DFT transforms discrete K-element sequence of samples from time domain $x(k)$ into its equivalent in frequency domain $X(m)$:

$$X(m) = \sum_{k=0}^{K-1} x(k) \mathrm{e}^{-\frac{j2\pi km}{K}} \tag{2.9}$$

Result of this transformation is a periodic K-element sequence of values $X(m)$ distributed evenly along frequency axis at points:

$$f(m) = m \cdot \frac{f_{sample}}{K} \tag{2.10}$$

where f_{sample} is sampling frequency of $x(k)$. Obtaining correct results depends upon choosing proper input signal frequency f_{in} – it should be related to sampling frequency f_{sample} and number of collected samples K by relation [10, 16]:

$$f_{in} = \frac{J}{K} f_{sample} \tag{2.11}$$

where J is mutually prime number to K. If this condition is not met, spectral leakage will occur – input signal will be spread among whole frequency range $f(m)$ instead of one point, which will lead to false results of DFT.

Fourier transform can be used to analyse signals constituted of many components (e.g. signal and its harmonic) thanks to its linearity – transform of sum of signals is equivalent to sum of transformed signals [15]:

$$\begin{aligned} X_{sum}(m) &= \sum_{k=0}^{K-1} \left[x_1(k) + x_2(k)\right] \mathrm{e}^{-\frac{j2\pi km}{K}} = \sum_{k=0}^{K-1} x_1(k) \mathrm{e}^{-\frac{j2\pi km}{K}} + \sum_{k=0}^{K-1} x_2(k) \mathrm{e}^{-\frac{j2\pi km}{K}} \\ &= X_1(m) + X_2(m) \end{aligned} \tag{2.12}$$

Total Harmonic Distortion (THD)

Harmonics distortions at ADCs output (presence of harmonics of input signal) appear due nonlinearities in converter and due to missing codes. To measure how much those distortions degrade ADC performance parameter THD was defined as ratio of total power of harmonics to power of fundamental signal [10, 16]:

$$THD = 20 \log_{10} \left(\sqrt{\frac{\sum_{k=2}^{K_H+1} X_{avrg}^2((k \cdot f_{sig}) \bmod f_{sample})}{X_{avrg}^2(f_{sig})}} \right) \qquad (2.13)$$
$$[THD] = dB$$

where K_H is number of harmonics taken into account (usually $K_H = 10$) and $X_{avrg}(i)$ is average of measured values of frequency interval $f(i)$.

Signal to Non-Harmonic Distortion (SNHR)

To measure influence of error sources other than harmonics, e.g. quantization noise and noise introduced by capacitances and resistances, SNHR is used. It is defined as power of signal to total power of all other frequency intervals within measured bandwidth, excluding harmonics frequencies [10, 16]:

$$SNHR = 20 \log 10 \left(\sqrt{\frac{X_{avrg}^2(f_{sig})}{\sum_{k=1}^{2^K-1,\ i \neq f_{h[k]}} X_{avrg}^2(f(i))}} \right) \qquad (2.14)$$
$$[SNHR] = dB$$

where $f_{h[k]} = (k \cdot f_{sig}) \bmod f_{sample}$.

Signal to Noise and Distortion (SINAD)

Dynamic parameter which describes overall performance of ADC is SINAD – here both harmonic and nonharmonic sources of noise are taken into account. SINAD is defined as ratio of power of signal to total power of noise and distortion within measured bandwidth:

$$SINAD = 20 \log 10 \left(\sqrt{\frac{X_{avrg}^2(f_{sig})}{\sum_{k=1}^{2^K-1,\ i \neq f_{sig}} X_{avrg}^2(f(i))}} \right) \qquad (2.15)$$
$$[SINAD] = dB$$

Effective Number Of Bits (ENOB)

This parameter was already defined in section 2.1.1 among static parameters, but it can be calculated also based on dynamic parameters. Equation for ENOB is based on definition of *Singal to Noise Ratio (SNR)* for ideal ADC. SNR is defined as [14, 16]:

$$\begin{aligned} SNR &= \log_{10}\left(\frac{Power\ of\ signal}{Power\ of\ noise}\right) \\ [SNR] &= dB \end{aligned} \tag{2.16}$$

Standard input signal is a sinus wave with amplitude of $\frac{V_{ref}}{2}$, thus its average power $\langle P_{sin}\rangle$ can be calculated as (using very common simplification $R = 1\Omega \Rightarrow P = \frac{V^2}{R} = V^2$):

$$\langle P_{sin}\rangle = \frac{1}{T}\int_0^T \left(\frac{V_{ref}}{2}\sin(2\pi f t)\right)^2 \mathrm{d}t = \frac{V_{ref}^2}{8} \tag{2.17}$$

To calculate noise power we assume that all components are ideal, so only quantization noise is present (this assumption poses some restrictions on quantization process – quantization levels must be uniform, equiprobable, not correlated to input and large number of them must exist [14]; all those conditions are met for an ideal 12-bit ADC with high swing input signal). From Figure 2.2b for ideal ADC it is clear that $\epsilon_Q \in \left[-\frac{LSB}{2}; \frac{LSB}{2}\right]$, and additionaly we assume that probability distribution of quantization error $P(\epsilon_Q)$ is constant within mentioned range and equal to zero outside it. Probability normalization leads to: $\int_{-\infty}^{\infty} P(\epsilon_Q)d\epsilon_Q = 1 \Rightarrow P(\epsilon_Q) = \frac{1}{LSB}$. All this allows to calculate the average noise power as:

$$\langle P_{noise}\rangle = \int_{-\infty}^{\infty} P(\epsilon_Q)\epsilon_Q^2 \mathrm{d}\epsilon_Q = \frac{1}{LSB}\int_{-\frac{LSB}{2}}^{\frac{LSB}{2}} \epsilon_Q^2 \mathrm{d}\epsilon_Q = \frac{LSB^2}{12} \tag{2.18}$$

Combining equations 2.1, 2.16, 2.17 and 2.18 results in:

$$SNR_{ideal\ ADC} = 10\log_{10}\left(\frac{3}{2}\cdot 2^N\right) \approx 6,02\cdot N + 1,76\ [dB] \tag{2.19}$$

Expression 2.19 shows highest achievable SNR for N-bit converter. For comparison – if an ADC would have a transfer characteristics like "simple ADC" from figure 2.2a than its noise power would be $\langle P_{noise\ simple}\rangle = \frac{1}{LSB}\int_0^{LSB} \epsilon_Q^2 \mathrm{d}\epsilon_Q = \frac{LSB^2}{3}$, hence

$$SNR_{simple\ ADC} = 10\log_{10}\left(\frac{3}{8}\cdot 2^N\right) \approx 6,02\cdot N - 4,26\ [dB] \tag{2.20}$$

Calculating value of N from equation 2.19:

$$N = \frac{SNR - 1.76}{6.02} \tag{2.21}$$

By substituting $ENOB$ for N and $SINAD$ for SNR commonly used equation for $ENOB$ is obtained [16]:

$$ENOB = \frac{SINAD - 1.76}{6.02} \tag{2.22}$$

2.2 Overview of ADC architectures

One of most important theorems in signal conversion is Nyquist-Shannon theorem [14]:

> *A band limited signal $x(t)$, which Fourier spectrum $X(j\omega)$ vanishes for frequencies $|f| < \frac{1}{2} f_{sample}$ is fully described by a uniform sampling $x(\frac{n}{f_{sample}})$, where $n \in \mathbb{N}$.*

Based on this theorem all existing architectures of analog-to-digital converters can be divided into two categories:

- Nyquist rate ADC – input signals have maximal frequency twice (or little bit more) lower than that of sampling. This category is represented by many different architectures, some of which are:
  - Flash converter
  - Pipeline converter
  - Successive approximation converter
- oversampling ADC – frequency of sampling is many times higher than that of input signals. Only $\Sigma - \Delta$ converters works in this way.

Each architecture is suitable for different purpose depending on number of bits, sampling frequency, power and area consumption needed (as presented in Figure 2.4). Following section will give a brief summary of each mentioned architecture.

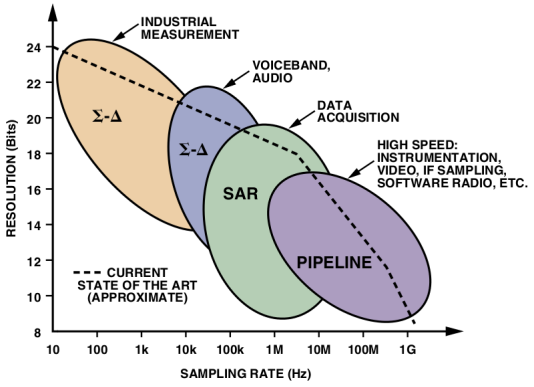


Figure 2.4: General allotment of different architectures of ADC. [17]

2.2.1 Oversampling ADC

Oversampling ADCs work very differently compared to Nyquist rate converters – they relay on noise shaping and oversampling followed by averaging of input signal. They are capable of achieving very high resolution (even 24-bits) but maximal input signal frequency rarely exceeds few MHz. For those reasons they are mostly used for processing of sound. Operations of $\Sigma - \Delta$ converter (only kind of oversampling ADC) is easiest to summarize by explaining function of each of its building blocks (block diagram of converter is shown in Figure 2.5):

- Antialiasing filter – filters out any noise outside signal bandwidth, so it will not be aliased back close to signal during oversampling
- Sampling circuit – samples input signal with frequency many times higher than frequency resulting from Nyquist-Shannon theorem
- Modulator – many different kinds of modulators are used varying mainly in number of incorporated integrators and resolutions of analog-to-digital and digital-to-analog converters (not necessarily having the same resolution), but simplest one can for example consist of one integrator, comparator as ADC and switch as DAC connected in one feedback loop. This block has two main functions:
  - shaping noise in such a way that majority of it is shifted to high frequencies
  - producing at the output a digital signal with frequency equal to sampling frequency and mean value equal to value of sampled input
- Digital filter – most commonly a low-pass filter, which should remove noise shifted to high frequencies
- Decimator – produces a lower frequency (compared to sampling frequency) converter output signal by averaging filtered modulator output over set period of time. Simplest implementation is a counter which counts number of pulses over pre-set number of cycles.

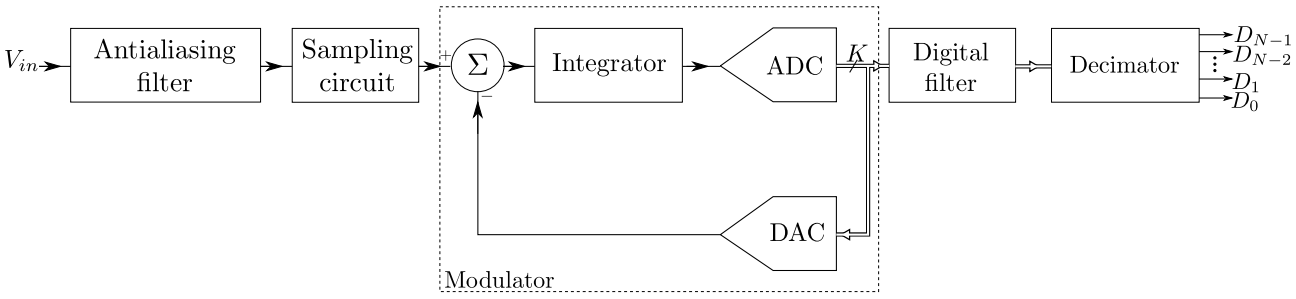


Figure 2.5: Block diagram of $\Sigma - \Delta$ ADC.

2.2.2 Nyquist rate ADC

Flash converter

Principle of operation of flash ADC is simple – input voltage is compared with every transition point of ADC at the same time, resulting in information on how many LSBs are required to match sampled voltage level. This number is than translated into binary value by digital logic – thanks to such means of operation flash ADC is fastest of all ADC.

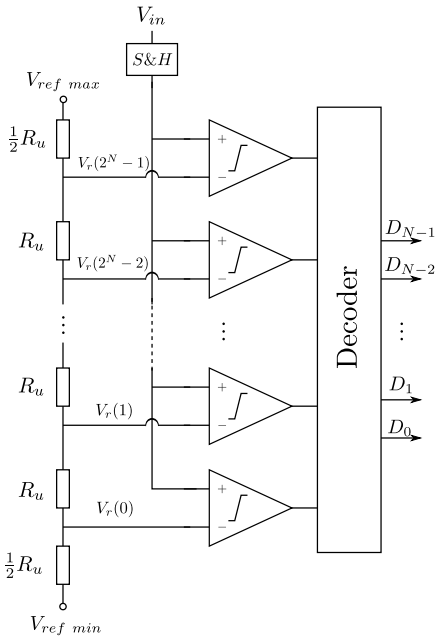


Figure 2.6: Schematic of simple Flash ADC.

Simple implementation using resistive divider is shown at Figure 2.6. First and last resistor have half of unit resistance value R_u to achieve transfer characteristic similar to ideal ADC from Figure 2.2a. In such configuration one of the inputs of each comparator is V_{in} and second one is $V_r(i)$:

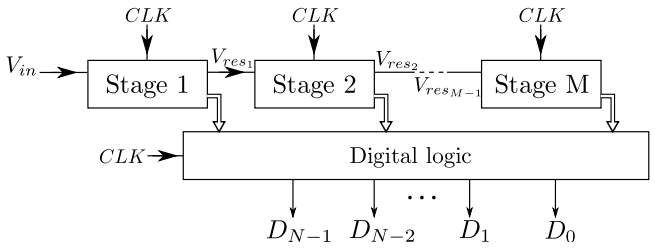
$$V_r(i) = \frac{i + \frac{1}{2}}{2^N - 1} \left(V_{ref\ max} - V_{ref\ min}\right) + V_{ref\ min} \tag{2.23}$$

The way this converter works limits its usage to relatively low resolutions due to various reasons:

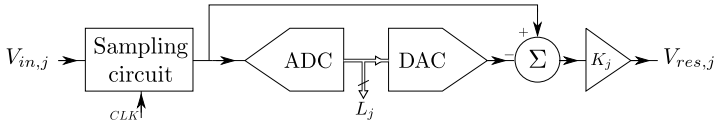
- rise in resolution by 1 bit requires twice more transition points hence twice smaller resistances in divider – scaling those down to very small values introduces very strong influence of mismatch and because of that becomes impossible for high resolution
- number of required comparators is $2^N - 1$, resulting in exponential growth of power consumption and area of converter
- offset of all comparators needs to be kept below $\frac{1}{2}$ LSB, resulting in very small and hard to achieve values for high resolutions

Pipeline converter

Pipeline converter is build out of cascade of individual stages (not necessarily identical) each performing part of conversion (simple block diagram of pipeline ADC is presented in Figure 2.7).



(a) Block schematic of pipeline ADC.



(b) One of stages in pipeline converter.

Figure 2.7: Pipeline architecture.

Pipeline ADC's sequence of conversion starts with input signal being sampled by 1^{st} stage (as shown in Figure 2.7b). This sample is converted to digital value by an L-bit ADC (L can have any value, but most commonly low values are used, often single bit). This value serves as one output of stage – conversion result. ADCs output is than converted back to analog form, subtracted from original input and the result is amplified K_1 times producing residual value $V_{res_1} = (V_{in} - V_{DAC_1}) \cdot K_1$, or put in more general form for j-th stage;:

$$V_{res,j} = (V_{res,j-1} - V_{DAC,j}) \cdot K_j \tag{2.24}$$

The residual value $V_{res,j}$ serves as input for next stage. Gain factor K_j is often set to be 2^{L_j} so all stages can use the same reference voltage. Output of each stage is passed to digital logic, which after last stage finishes conversion forms digital output word based on results of stages conversions. For converter build out of M stages it takes $M + 1$ clock cycles to convert signal (assuming that digital logic operation takes only one cycle), but advantage of this architecture is that after given stage has done conversion for one sample it can immediately start conversion for another one – in such mode of operation after initial wait of $M + 1$ clock cycles conversion results are provided every clock cycle (with the same resolution) despite the fact that conversion itself always takes $M + 1$ cycles. Pipeline ADC achieve medium resolution (8 to 12 bits) and consume moderate amounts of power – for those reasons they were very commonly used but in recent years are being superseded by SAR ADC.

Succesive Approximation Register (SAR) converter

Schematic of simple SAR ADC is shown on Figure 2.8. It consists of sample and hold circuit, comparator, SAR control logic and DAC (used to produce reference voltage for comparator). Conversion starts by sampling input signal (sampling duration is controled by CLK_{sample}) at the end of which the logic set Most Significant Bit (MSB) to 1 causing DAC's output voltage V_{DAC} to rise to $\frac{1}{2}V_{ref}$. Than comparator decides which of those two voltages is higher:

- if $V_{in} > V_{DAC}$ than the value of sampled voltage is higher than $\frac{1}{2}V_{ref}$, so first bit of output word was guessed corectly and remains 1
- if $V_{in} < V_{DAC}$ than the value of sampled voltage is lower than $\frac{1}{2}V_{ref}$, so first bit of output word was guessed incorectly and is reset to 0

After this check next bit (MSB-1) is changed to 1 (resulting in $V_{DAC} = \frac{3}{4}V_{ref}$ if MSB = 1 or $V_{DAC} = \frac{1}{4}V_{ref}$ if MSB = 0) and whole process is repeted until all N bits are resolved – this algorithm can be summarized as shown on Figure 2.9a and example waveform is shown on Figure 2.9b.

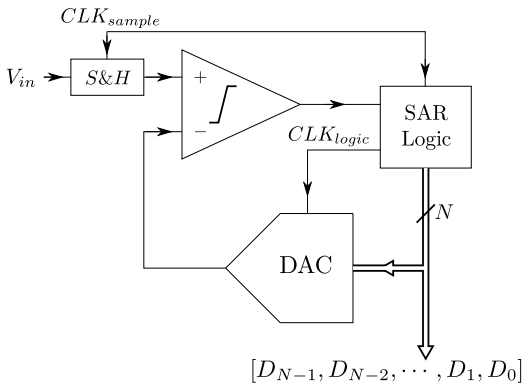
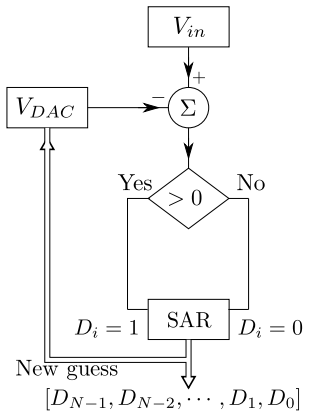
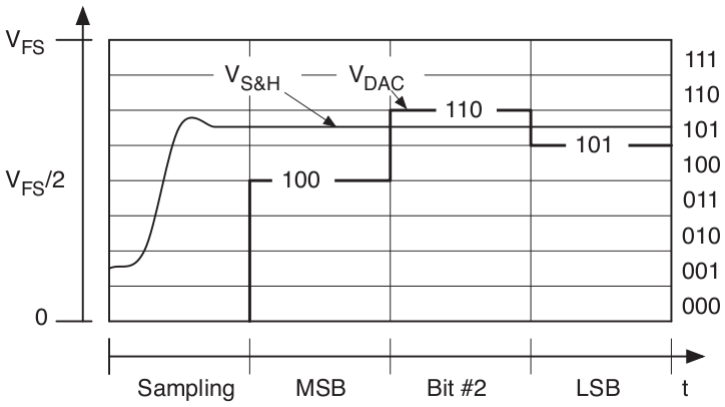


Figure 2.8: Block schematic of simple SAR ADC.



(a) Simple SAR ADC algoritm.



(b) Waveform of sampled signal (thin line) and SAR ADC approximation (thick line). [14]

Figure 2.9: Succesive approximation algorithm and example of 3-bit conversion.

Although this way of approximation works correctly it is quite wasteful in regard of power e.g. if $V_{in} = V_{ref\ min}$ DAC is charged and discharged for every bit wasting energy. For this reason SAR architecture was not very popular until few years ago when improved implementations of successive approximation algorithm were proposed – those, alongside with more detailed examination of SAR ADC will be described in the next chapter.

3 | SAR ADC – algorithm variants and building blocks general considerations

3.1 Variants of successive approximation algorithm

As was mentioned in previous chapter recent years brought many developments in successive approximation ADCs. This evolution of SAR architecture is mainly driven by need for medium resolution ultra low-power ADCs and is made possible by advances of technology used to manufacture integrated circuits and optimization of SAR ADC architecture.

Despite the fact that from the point of view of basic functionality (providing voltage reference) any DAC architecture can be used in SAR ADC (resistor string, R-2R ladder, current steering, etc.) most commonly used one is charge scaling DAC – all reviewed in this chapter successive approximation methods use this kind of digital-to-analog converter. In its simplest implementation (presented in Figure 3.1) it consists of a parallel array of binary-weighted capacitors, resulting in total capacitance C_{tot} of:

$$C_{tot} = \left(\sum_{i=0}^{N-1} 2^i + 1 \right) C_u = 2^N C_u \tag{3.1}$$

where C_u is unit capacitance. All capacitors are connected together by one plate while second plate of each capacitor is connected to a separate switch S_i providing a voltage level appropriate at the current conversion phase (in example shown in Figure 3.1 those are: reference voltage V_{ref} and ground level V_{gnd}). Output of such converter V_{DAC} is a result of voltage division among capacitors.

$$V_{DAC} = V_{ref} \cdot \sum_{i=0}^{N-1} S_i 2^{i-N} + V_{gnd} \tag{3.2}$$

The main differences between presented methods of successive approximation are architec-

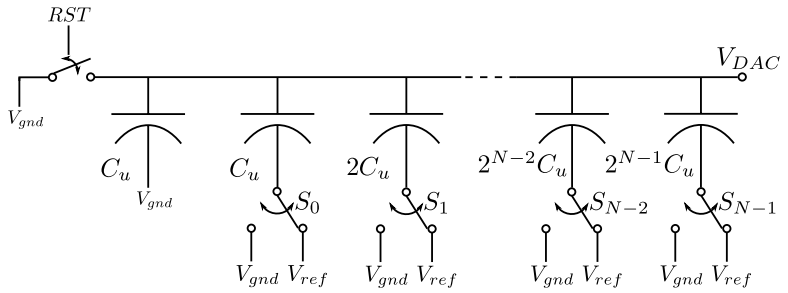


Figure 3.1: Schematic of simple charge scaling DAC.

ture of used DAC and algorithm of its switching, but no matter how an array of capacitors will be modified, the output value will always be a result of voltage division among capacitors. This short overview of charge scaling DAC should have provided enough information to allow understanding of concepts described in this chapter, a more detailed examination of this building block will be presented in chapter 3.2.

Following sections contain overview of some of the more popular variants of SAR algorithm, each illustrated with an example showing all possible states in all conversion stages for 3-bit ADC (except for improved switchback algorithm, were minimal example showing all techniques requires 4-bits). Values of energy marked in all those examples are values drawn from voltage source due to the operation of DAC and are noted over blue arrows showing transitions between stages. In all cases those values are calculated based on the change of the charge stored in capacitors connected to voltage source after switching is done. Assuming that transition between stages starts at T_1 and ends at T_2 the energy drawn from source $E_{T_1 \to T_2}$ can be calculated as [18]:

$$E_{T_1 \to T_2} = \int_{T_1}^{T_2} i_{source}(t) v_{source}(t) \mathrm{d}t = \left\| \begin{matrix} v(t) = V_{source} \\ i_{source}(t) = \frac{\mathrm{d}Q}{\mathrm{d}t} \end{matrix} \right\| = V_{source} \int_{T_1}^{T_2} \frac{\mathrm{d}Q}{\mathrm{d}t} \mathrm{d}t = V_{source} \int_{Q(T_1)}^{Q(T_2)} \mathrm{d}Q \tag{3.3}$$

By definition charge Q stored in capacitor is equal to product of its capacitance C_{cap} and voltage across it V_{cap}, leading to:

$$E_{T_1 \to T_2} = C_{cap} V_{source} \left[V_{cap}(T_2) - V_{cap}(T_1) \right] \tag{3.4}$$

Since the design presented in this work is fully differential all reviewed methods will be also shown in differential configuration. Based on equation 3.4 calculations of energy consumption for all described algorithms were implemented in Matlab 2009b and results are presented throughout this chapter as energy consumption plots. Code itself is included as appendix C.

3.1.1 Classical algorithm

In the classical approach [18, 19] (presented in Figure 3.2) two capacitive DACs work in complementary way to converge DAC top plate voltages after sampling to $V_{cm} = \frac{V_{ref}+V_{gnd}}{2}$. In sampling phase input is sampled on bottom plate of capacitive DACs while top plates are being charged to common-mode voltage V_{cm}. When the sampling ends the top plates are disconnected from V_{cm} and all bottom plates of DAC capacitive network sampling $V_{in,p}$ are connected to V_{gnd}, except for biggest capacitor $2^{N-1}C_u$ which is switched to V_{ref} (voltage at comparator input for this side is then $V_{cm} - V_{in,p} + \frac{1}{2}V_{ref}$). Bottom plates of DAC network sampling $V_{in,n}$ are switched in a complementary way – $2^{N-1}C_u$ is connected to V_{gnd}, while the rest is switched to V_{ref}. Then the first comparison is performed – D_{N-1} which is the Most Significant Bit (MSB) is decided and based on that decision bottom plate voltages of MSB capacitors ($2^{N-1}C_u$) are set:

- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, resulting in $S_{N,p} \rightarrow V_{ref}$ and $S_{N,n} \rightarrow V_{gnd}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, resulting in $S_{N,p} \rightarrow V_{gnd}$ and $S_{N,n} \rightarrow V_{ref}$

After that the second largest capacitors bottom plates are switched (on $V_{in,p}$ sampling side to V_{ref}, on $V_{in.n}$ sampling side to V_{gnd}) and the whole process is repeated. Conversion ends when all bits have been resolved. Every bit found brings the difference between two DACs top plate voltages ΔV_{DAC} closer to V_{cm} – after the last bit is found this difference should be $|\Delta V_{DAC}| \leq LSB$.

Although this approach to successive approximation is quite intuitive it is also wasteful in respect to power consumption, especially when wrong assumptions are made – this can be seen from Figure 3.3 (energy used for DAC switching for output code $0(00\cdots00)$ is much higher than for $4095(11\cdots11)$). Overview of most important features of this algorithm is presented in Table 3.1.

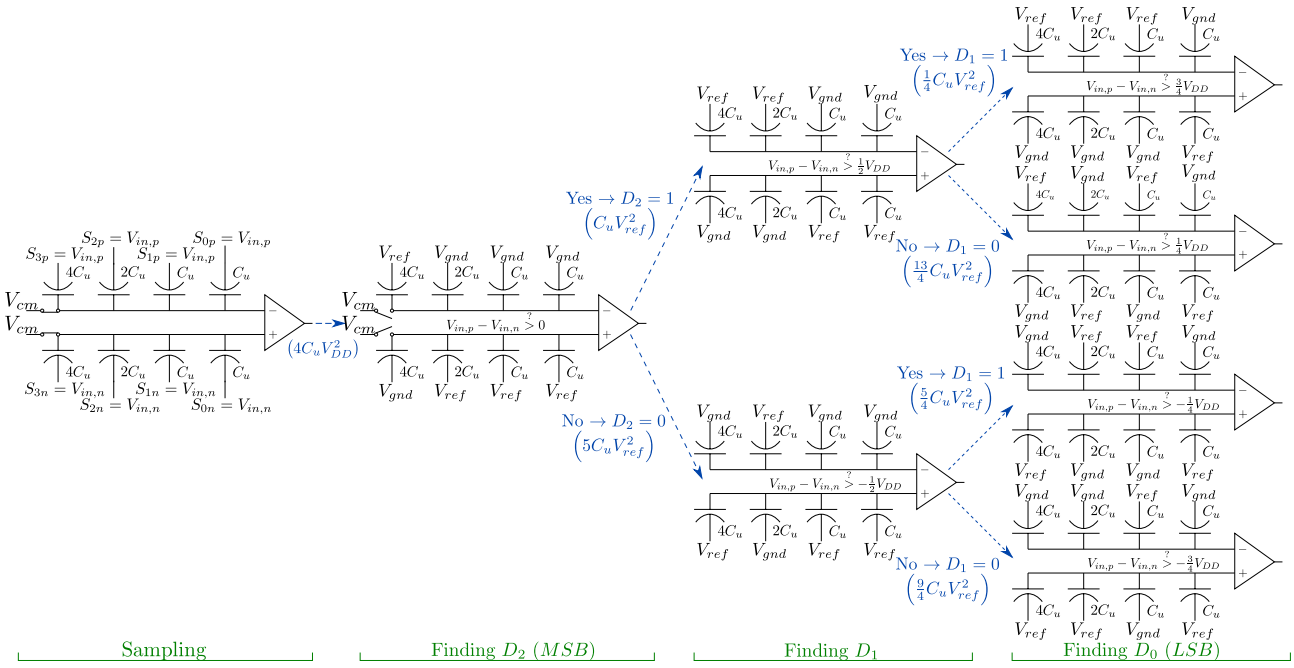


Figure 3.2: 3-bit SAR ADC incorporating classical algorithm.

Calculations leading to energy values shown on Figure 3.2 are presented in appendix A. For all other algorithms such calculations can be carried out in the same manner – for this reasons full calculations will be omitted and only results will be presented.

Required DAC resolution (for N-bit ADC)	N
Number of needed C_u (for differential DAC)	$2 \cdot 2^N$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	$\frac{1}{2}V_{ref}$

Table 3.1: Features of classical algorithm.

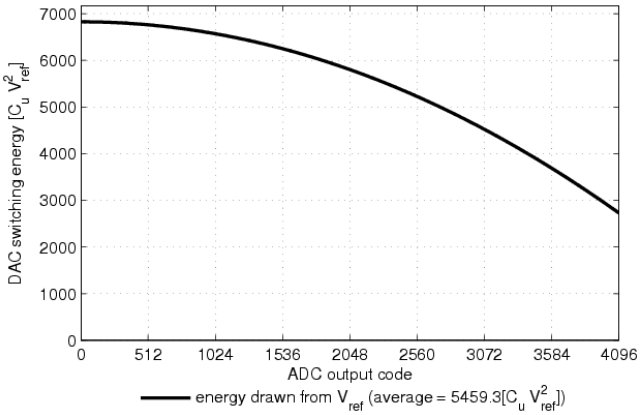


Figure 3.3: Energy consumption due to DAC switching for 12-bit ADC using classical algorithm.

3.1.2 Energy saving

Energy saving algorithm [20] uses modified DAC architecture – second biggest capacitor is split into binary divided array in which each of scaled capacitors can be switched separately. Such configuration allows to share part of charge accumulated in this sub-DAC instead of just discharging $2^{N-2}C_u$ to ground as happens in classical algorithm. Operations of positive

and negative DACs are complementary, so for simplicity only positive DAC switching will be described.

Energy saving algorithm (example of 3-bit ADC using this method is presented in Figure 3.5) implements bottom plate sampling (top plates are held at V_{ref}). After sampling is done, top-plates are disconnected from V_{ref} and all bottom plates are switched to V_{gnd}. In such state first comparison is made, resulting in finding value of D_{N-1} and next configuration of DAC's bottom-plate voltages:

- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, leading to $\forall_i S_{N-2,i,p} \rightarrow V_{gnd}$ and $\forall_{j \neq N-2} S_{j,p} \rightarrow V_{ref}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, leading to $\forall_i S_{N-2,i,p} \rightarrow V_{ref}$ and $\forall_{j \neq N-2} S_{j,p} \rightarrow V_{gnd}$

If $D_{N-1} = 0$ than i-th decision that is 1 will result in connecting capacitance $2^{N-2-i}C_u$ of sub-DAC to V_{ref} and $D_{N-1-i} = 0$ will lead to disconnecting 2^{N-2-i} capacitance from main DAC from V_{ref} and connecting it to V_{gnd}. If D_{N-1} was 1 than operations will be complementary. Figure 3.5 presents an example of 3-bit ADC using this algorithm.

Although this algorithm is more energy efficient than classical one (by 56.25% as can be seen from Table 3.2 or by comparing Figures 3.3 and 3.4), splitting second largest capacitor in separate binary weighted array to be used during switching requires nearly double the number of switches in circuit – for high resolution ADC those switches and buffers needed to drive them might be quite big, resulting in additional area needed for layout.

Required DAC resolution (for N-bit ADC)	N
Number of needed C_u (for differential DAC)	$2 \cdot 2^N$
Required reference sources	V_{gnd}, V_{ref}
DAC's convergence voltage	$\frac{1}{2}V_{ref}$
Efficiency $(1 - E_{avg}/E_{avg,classic})$	56.25%
Remarks	Requires 2N-1 switches for each DAC

Table 3.2: Features of energy saving algorithm.

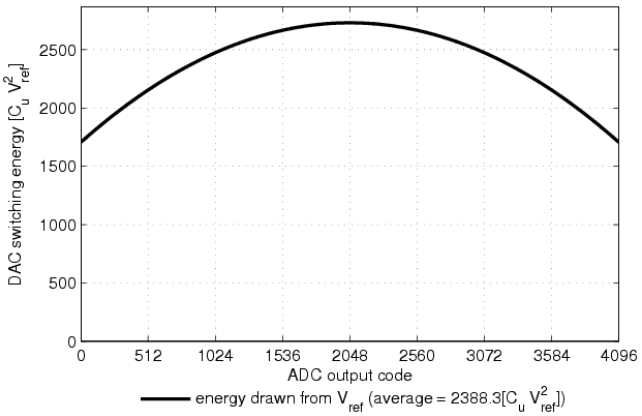


Figure 3.4: Energy consumption due to DAC switching for 12-bit ADC using energy saving algorithm.

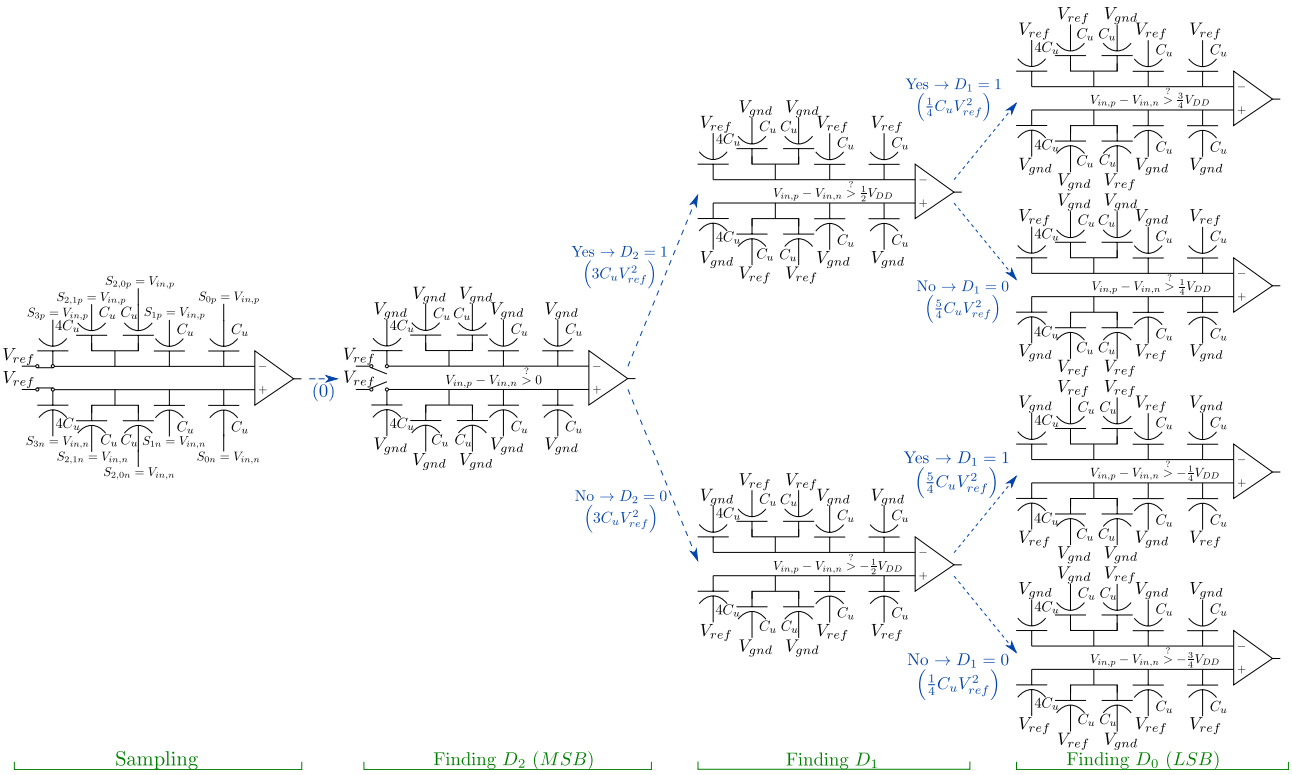


Figure 3.5: 3-bit SAR ADC incorporating energy saving switching algorithm.

3.1.3 Monotonic switching

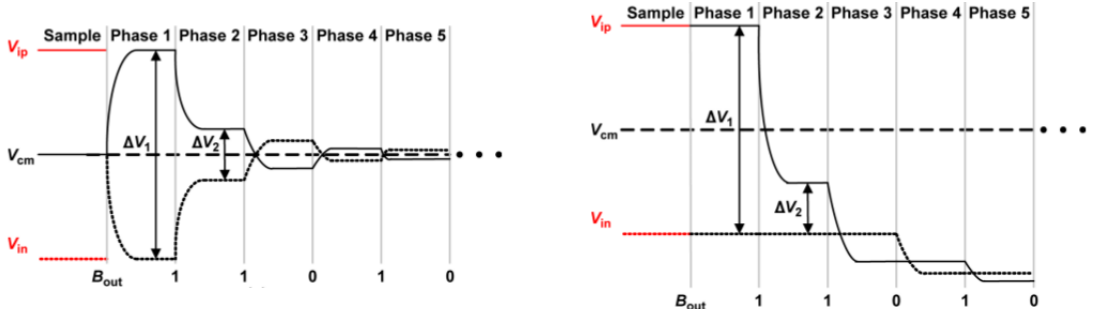
Other name used for this method is set-and-down algorithm [19, 21], idea behind it is to converge voltages sampled on DACs' top plates to V_{gnd} instead of converging them to V_{cm} as in classical method - the comparison of V_{DAC} voltages during conversion for classical and monotonic switching algorithms is presented in Figure 3.6.

During the sampling phase bottom plates of all capacitors are switched to V_{ref} while input is sampled onto top plates. After sampling is finished top plates are disconnected from input signal, bottom plates remain at V_{ref} and first comparison is performed. Based on decision of comparator D_{N-1} is resolved and appropriate bottom plate voltage is changed:

- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, resulting in $S_{N-2,p}$➛V_{gnd} and $S_{N-2,n}$➛V_{ref}
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, resulting in $S_{N-2,p}$➛V_{ref} and $S_{N-2,n}$➛V_{gnd}

After that the procedure is repeated until whole digital output word is found (Figure 3.8 shows an example of 3-bit ADC using monotonic switching).

In an alternative version of this algorithm the sampled voltages are converged to higher of the two. This method would use bottom plate voltages complementary to those described above (during sampling all capacitors would be switched to V_{gnd} and based on comparator decision one of capacitors would be switched to V_{ref} each cycle). Disadvantage of such approach is the need to switch bottom plate voltages from V_{gnd} to V_{ref} during conversion – this operation is slower than switching from V_{ref} to V_{gnd} for the same size of switch due to lower mobility of holes than electrons.



(a) Classical algorithm

(b) Monotonic switching

Figure 3.6: Comparison of voltage on DACs top plates during conversion using classical algorithm and monotonic switching. [19]

Required DAC resolution (for N-bit ADC)	$N-1$
Number of needed C_u (for differential DAC)	$2 \cdot 2^{N-1}$
Required reference sources	V_{gnd}, V_{ref}
DAC's convergence voltage	$\frac{1}{2}V_{ref} \rightarrow V_{gnd}$
Efficiency $(1 - E_{avg}/E_{avg,classic})$	81.25%
Remarks	Variable DACs common-mode

Table 3.3: Features of monotonic algorithm.

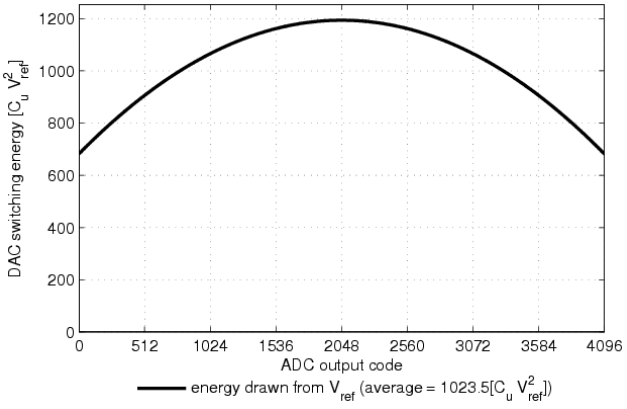


Figure 3.7: Energy consumption due to DAC switching for 12-bit ADC using monotonic algorithm.

Although monotonic switching algorithm uses half the number of unit capacitors compared to classical method (thanks to top-plate sampling required DAC's resolution can be lowered by 1 bit) and is more efficient energy-wise (by 81.25% as can be observed by comparing Figure 3.7 and 3.3) it has big disadvantage – sampled voltages common-mode gradually decreases from V_{cm} to V_{gnd}. This forces the comparator to work with very wide range of common mode degrading its performance. Overview of features of this algorithm is presented in Table 3.3.

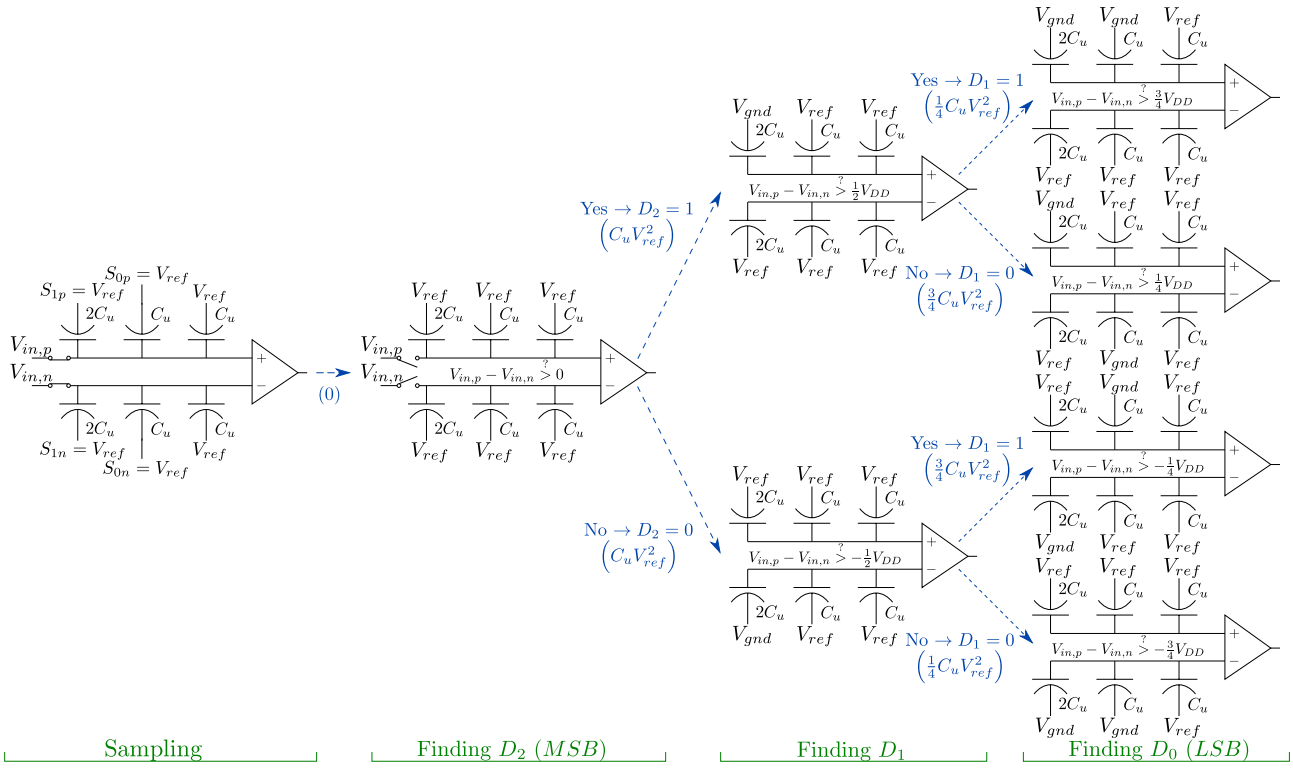


Figure 3.8: 3-bit SAR ADC incorporating monotonic switching algorithm.

3.1.4 Merged capacitor switching (MCS)

This algorithm (also called V_{cm}-based algorithm) [21, 22] uses top-plate sampling and three bottom-plate voltage levels to approximate sampled signal by converging voltage on both DACs to V_{cm}. Sampling starts with all bottom plates set to V_{cm} and sampling on top plates of capacitors. End of this phase results in switching off sampling switches and performing first comparison to find value of D_{N-1} (bottom plates remain at common mode voltage V_{cm}):

- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, leading to $S_{N-2,p} \rightarrow V_{gnd}$ and $S_{N-2,n} \rightarrow V_{ref}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, leading to $S_{N-2,p} \rightarrow V_{ref}$ and $S_{N-2,n} \rightarrow V_{gnd}$

After appropriate bottom plate voltages are found next comparison is performed following the same rules (as seen in example presented in Figure 3.10).

This method requires 1 bit lower DAC resolution compared to classical method (leading to lower number of capacitors needed) and is much more power efficient – comparison of Figure 3.9 and 3.3 shows 87.5% lower average energy consumption. Although additional reference voltage source V_{cm} is required, it does not need to be very accurate – it's actual voltage level does not influence differential DAC's output, it only decides DACs common-mode voltage value. Additionally no power is drawn from this source, as can be seen at Figure 3.9.

Required DAC resolution (for N-bit ADC)	$N-1$
Number of needed C_u (for differential DAC)	$2 \cdot 2^{N-1}$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	V_{cm}
Efficiency $(1 - E_{avg}/E_{avg,classic})$	87.5%
Remarks	V_{cm} voltage does not need to be accurate

Table 3.4: Features of MCS algorithm.

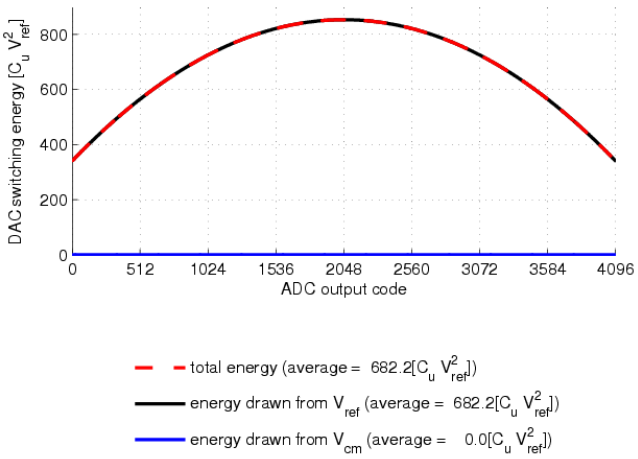


Figure 3.9: Energy consumption due to DAC switching for 12-bit ADC using MCS algorithm.

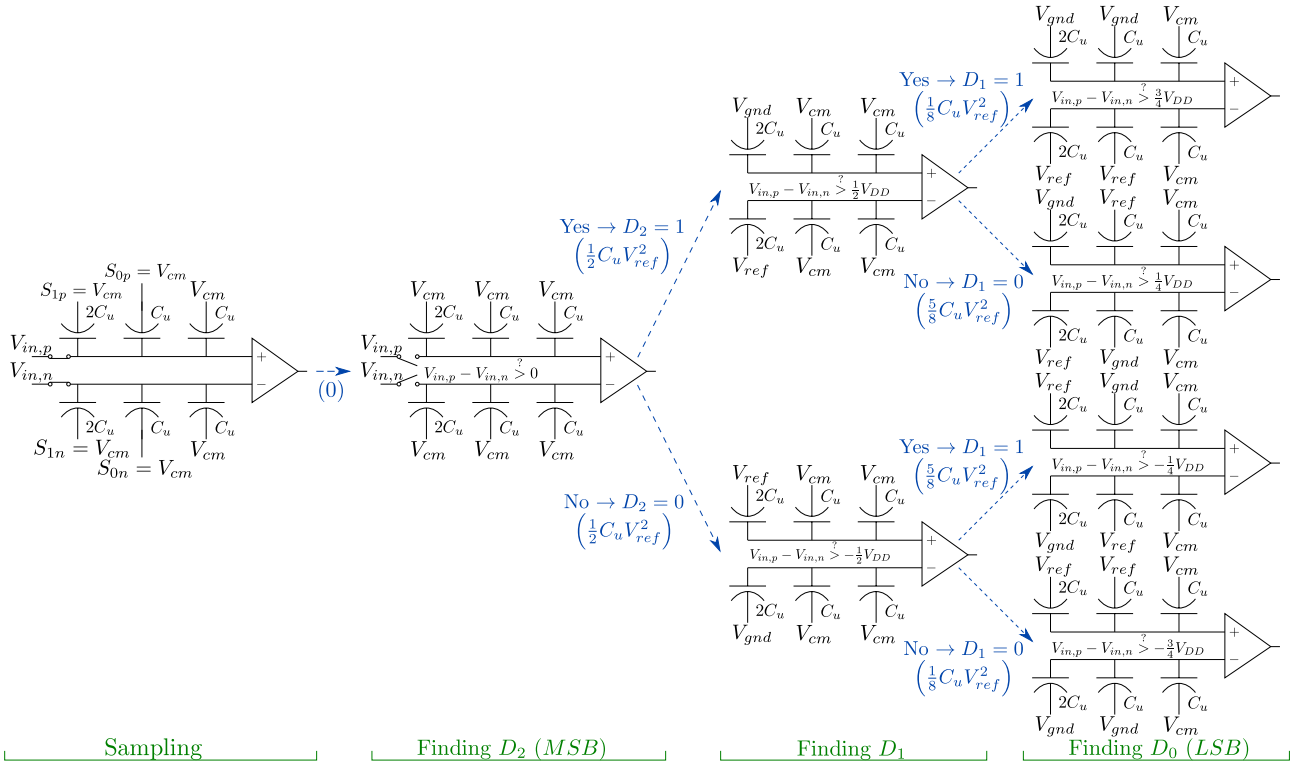


Figure 3.10: 3-bit SAR ADC incorporating merged capacitor switching (MCS) algorithm.

3.1.5 Early reset merged capacitor switching (EMCS)

This algorithm is an improvement of MCS algorithm focused on lowering energy consumption due to DAC switching – still top-plate sampling and three bottom-plate voltage levels are used to approximate sampled signal by converging voltage on both DACs to V_{cm}, but switching sequence is slightly modified [23].

Sampling starts with all bottom plates set to V_{cm} and sampling on top plates of capacitors.

End of this phase results in switching off sampling switches and performing first comparison to find value of D_{N-1} (bottom plates remain at common mode voltage):

- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, leading to $S_{N-2,p} \rightarrow V_{gnd}$ and $S_{N-2,n} \rightarrow V_{ref}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, leading to $S_{N-2,p} \rightarrow V_{ref}$ and $S_{N-2,n} \rightarrow V_{gnd}$

Further DAC switching follows the same rules as above if $S_{N-i,p}$ is supposed to be switched to the same voltage to which $S_{N-i-1,p}$ is connected. Otherwise (if switching according to rules above would result in connecting those two switches to different voltages), (as seen in example presented in Figure 3.14), $S_{N-i-1,p}$ is connected to V_{cm} and $S_{N-i,p}$ is connected to:

- V_{ref} if it was supposed to be connected to V_{gnd}
- V_{gnd} if it was supposed to be connected to V_{ref}

Those two switching phases (switching to V_{cm} and switching to V_{gnd} or V_{ref}) should be done one after the other (in any order), because simultaneous switching would reduce energy efficiency back to level of MCS algorithm, as presented in Figure 3.12.

A big advantage of described method is reducing INL and DNL by removing the worst case code switching such as $[10\cdots00] \rightarrow [01\cdots11]$. Overall effect on INL is presented in Figure 3.11. Furthermore this method requires 1 bit lower DAC resolution compared to classical method (leading to lower number of capacitors needed) and is much more power efficient – comparison of Figure 3.13 and 3.3 shows 89.07% lower average energy consumption. Additional reference voltage source V_{cm} does not need to be very accurate – it's actual voltage level does not influence differential DAC's output, it only decides DACs common-mode voltage value. No power is drawn from this source, as can be seen at Figure 3.13. Overview of basic features of EMCS algorithm is presented in Table 3.5.

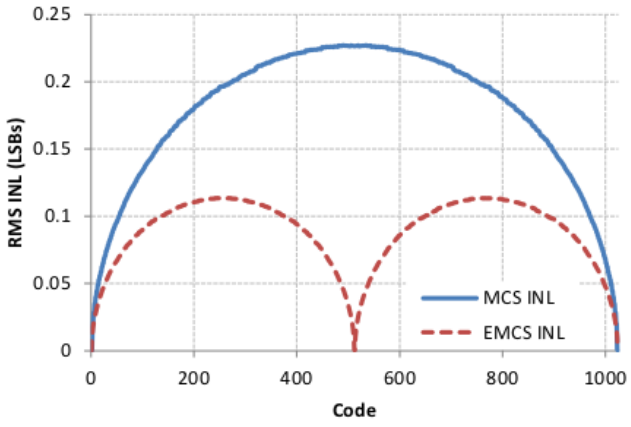


Figure 3.11: Comparison of INL for 10-bit ADCs using MCS and EMCS algorithm. [23]

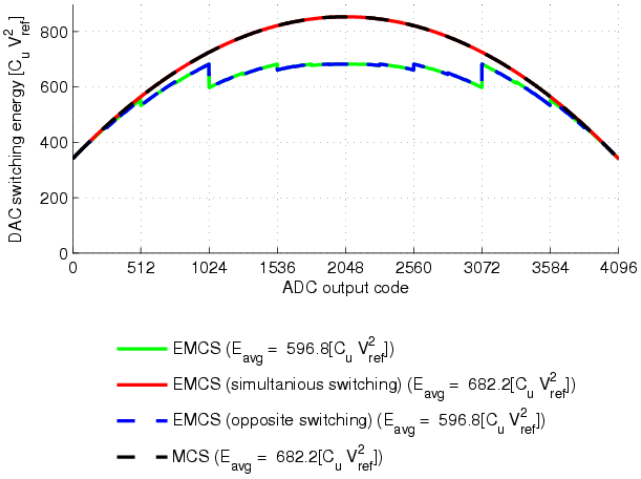


Figure 3.12: Energy consumption for different order of DAC switching in EMCS algorithm.

Required DAC resolution (for N-bit ADC)	$N-1$
Number of needed C_u (for differential DAC)	$2 \cdot 2^{N-1}$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	V_{cm}
Efficiency $(1 - E_{avg}/E_{avg,classic})$	89.07%
Remarks	each DAC switching done in two steps, low INL

Table 3.5: Features of EMCS algorithm.

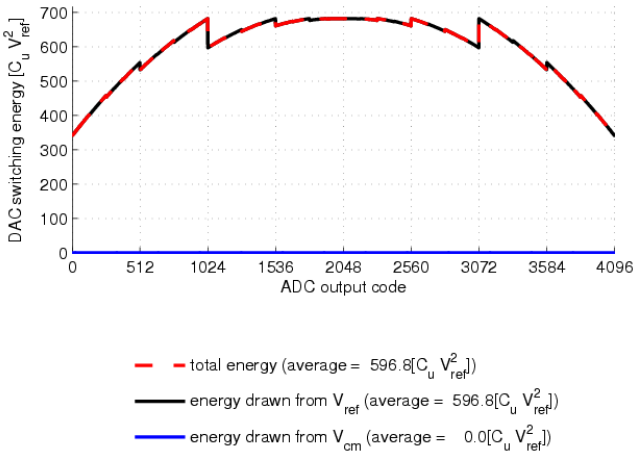


Figure 3.13: Energy consumption due to DAC switching for 12-bit ADC using EMCS algorithm.

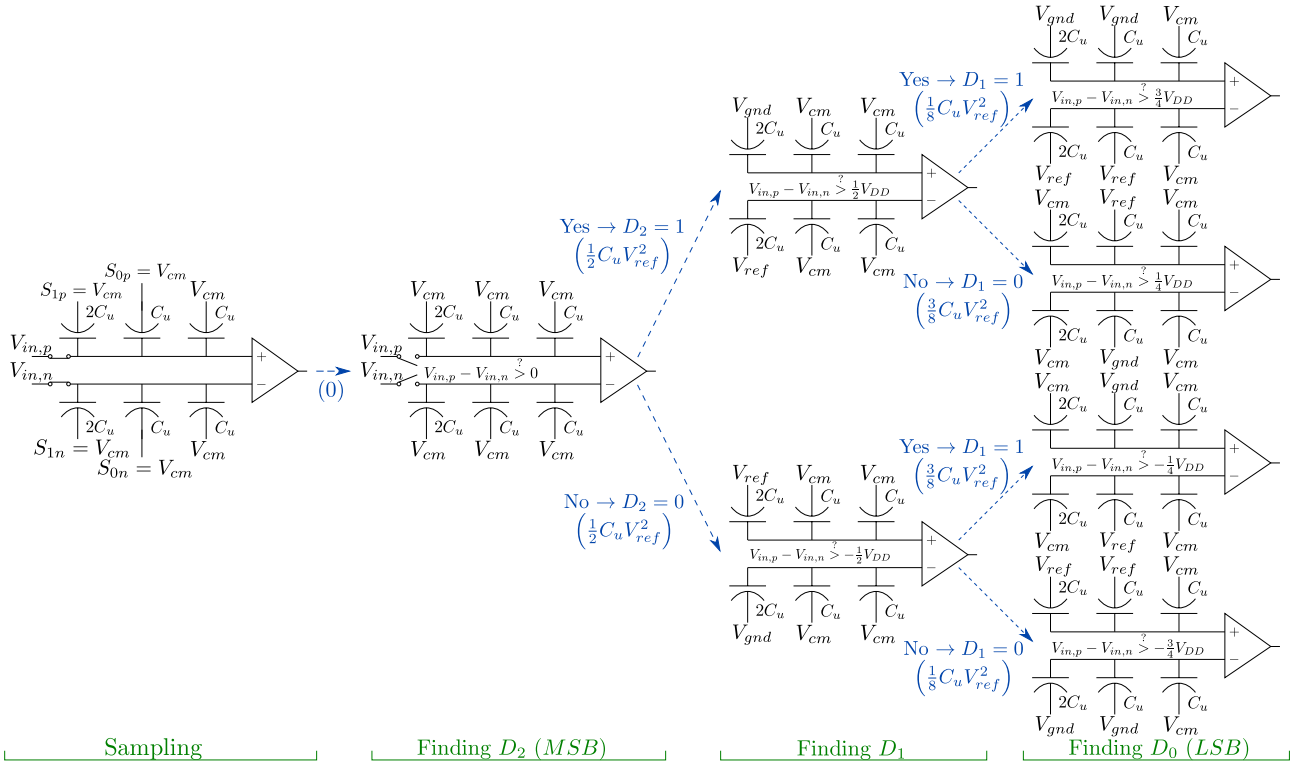


Figure 3.14: 3-bit SAR ADC incorporating early reset merged capacitor switching (EMCS) algorithm.

3.1.6 Asymmetric merged capactior switching (AMCS)

This method (described in [24, 25]) follows exactly the same algorithm as merged capacitor switching, except for last switching of capacitors – based on value of D_1 last pair of capacitor switches are set to:

- $D_1 = 1$ than $S_{0p} \rightarrow V_{gnd}, S_{0n} \rightarrow V_{cm}$
- $D_1 = 0$ than $S_{0p} \rightarrow V_{cm}, S_{0n} \rightarrow V_{gnd}$

One-sided switching causes the sampled voltage to converge to $V_{cm} - LSB$ rather than to V_{cm}, but for medium and high resolution ADCs this difference is very small and should not cause a problem for comparator (unlike large variations in common mode voltage observed in monotonic switching procedure). As presented in Figure 3.16 this method requires some modification to DAC:

- unit capacitor with bottom plate at fixed potential (used in other switching algorithm to ensure fully binary voltage scaling) is removed
- capacitances connected to S_1 and S_0 are the same size (in both DACs), rest of capacitances are scaled in usual way in respect to S_1 – this can be used either to lower the total number of used unit capacitors by half (when setting two smallest capacitance to C_u) or to improve matching (when setting them to $2C_u$)

This method requires 2-bit lower DAC resolution compared to classical method (leading to much lower number of capacitors needed) and has higher efficiency – comparison of Figure 3.15 and 3.3 shows 93.75% lower average energy consumption. Additional reference voltage source V_{cm} is required and it should provide an accurate voltage level since it's actual voltage value influence differential DAC's output in the LSB bit (on the other hand in case of inaccurate V_{cm} value an error will be introduced only in last bit). Additionally no power is drawn from this third source, as can be seen at Figure 3.15. Table 3.16 summarizes the features of AMCS algorithm.

Required DAC resolution (for N-bit ADC)	$N - 2$
Number of needed C_u (for differential DAC)	$2 \cdot 2^{N-2}$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	$V_{cm} - LSB$
Efficiency $(1 - E_{avg}/E_{avg,classic})$	93.75%
Remarks	Causes common-mode changes when used with split DAC (descr. in sect. 3.33)

Table 3.6: Features of AMCS algorithm.

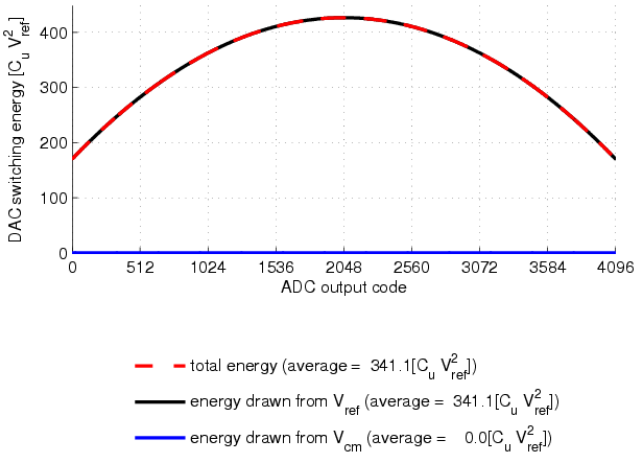


Figure 3.15: Energy consumption due to DAC switching for 12-bit ADC using AMCS algorithm.

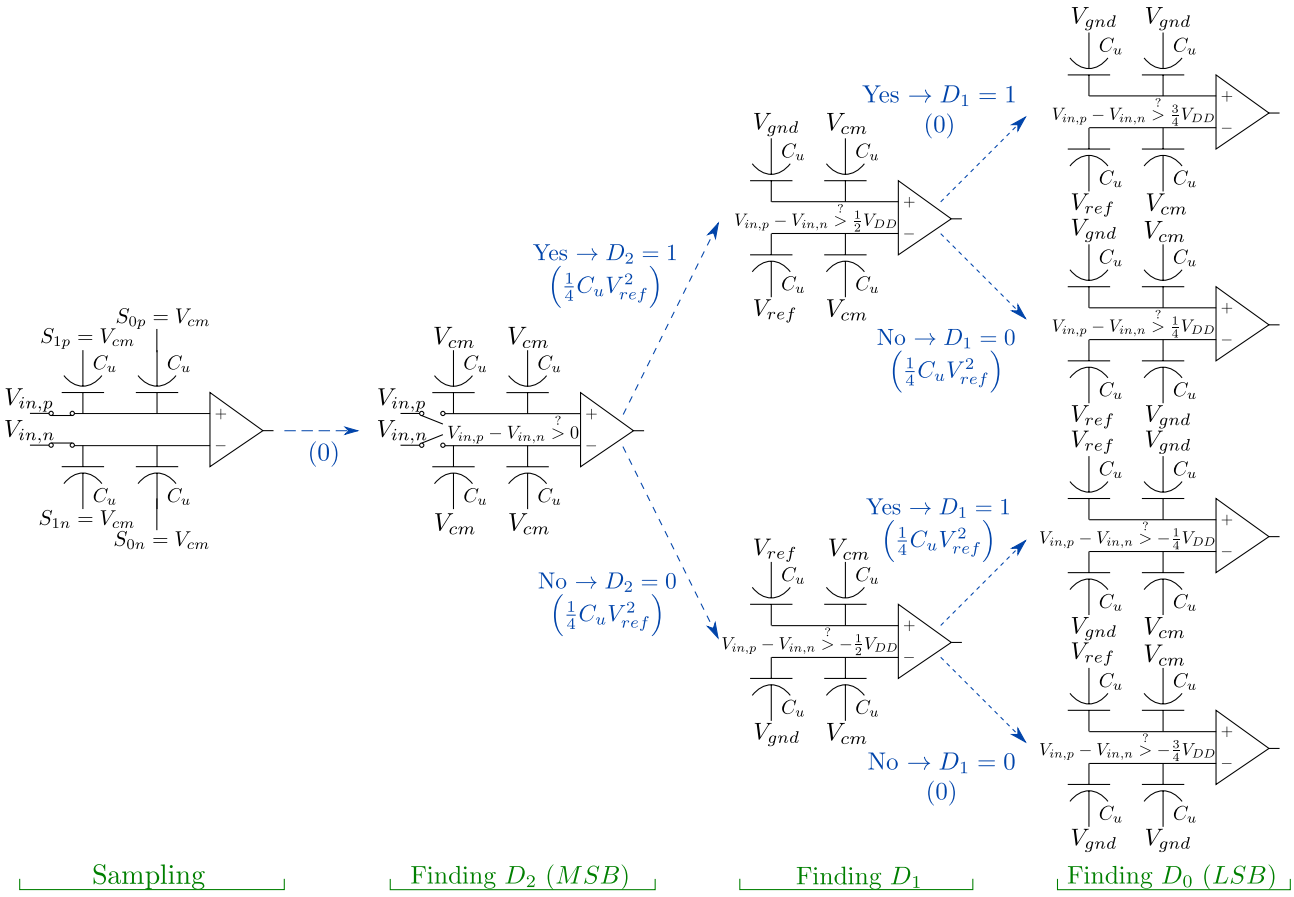


Figure 3.16: 3-bit SAR ADC incorporating asymmetric merged capacitor switching (AMCS) algorithm.

3.1.7 Tri-level switching

The idea behind this variation of successive approximation is based upon converging voltages sampled on DACs top plates to the voltage level of one of those samples (higher or lower one, depending on implementation) [26]. An example presenting 3-bit ADC is shown in Figure 3.19.

Conversion starts with sampling input on top plates of capacitive DACs, while all bottom-plates are kept at V_{gnd}. When sampling ends top plates are disconnected from input, bottom plates remain at low voltage level and first comparison is performed. Based on the result value of D_{N-1} and next step of approximation are decided:

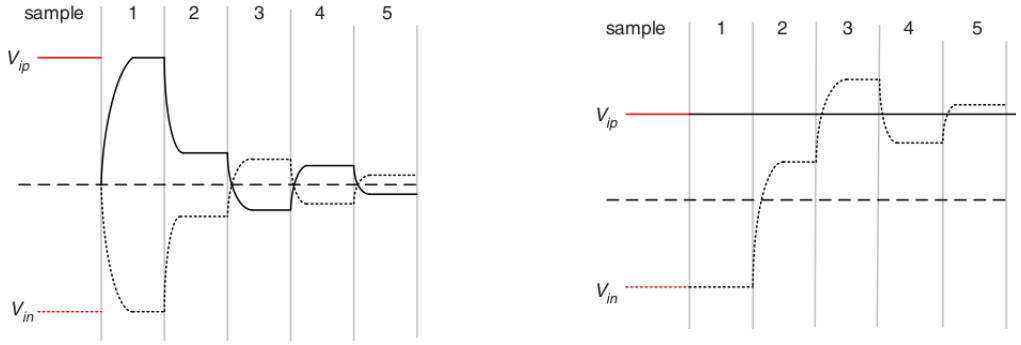
- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, resulting in $\forall_{i=0}^{2^{N-2}} S_{ip} \rightarrow V_{gnd}, S_{in} \rightarrow V_{cm}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, resulting in $\forall_{i=0}^{2^{N-2}} S_{ip} \rightarrow V_{cm}, S_{in} \rightarrow V_{gnd}$

Thus one of the DACs (the one with higher value of sampled voltage) remains in the same state as before comparison – this DAC will remain passive throughout the rest of conversion process, all switching will be done on second DAC (further referred to as active DAC). After all voltages are settled second comparison takes place and based on its result D_{N-2} and voltage of bottom plate of $2^{N-3}C_u$ of active DAC $S_{N-3,act}$ are decided:

- if $D_{N-2} = 1$ than $S_{N-3,act} \rightarrow V_{ref}$

– if $D_{N-2} = 0$ than $S_{N-3,act} \rightarrow V_{gnd}$

This ends second cycle of conversion. All further cycles follow the same methodology as use for finding D_{N-2}. This switching procedure results in variable DAC's convergence and common-mode levels (comparison between classical and tri-level algorithm waveforms is presented in Figure 3.17), which leads to the need of a comparator capable to work with wide range of common-mode voltages.



(a) Classical algorithm

(b) Tri-level switching

Figure 3.17: Comparison of voltage on DACs top plates during conversion using classical algorithm and tri-level switching. [26]

Although this method of conversion is very energy efficient (average switching energy is 96.87% lower than in classical method, as can be noticed from comparison of Figure 3.18 and 3.3) and requires 2-bit lower DAC resolution compared to classical method, an additional reference voltage source V_{cm} is required and it must provide a very accurate voltage level since it's voltage value influence differential DAC's output in all conversion phases. Power drawn from both reference sources for a 12-bit ADC is presented on Figure 3.18. Negative energy values on this figure mean that for given output code more energy is given back to source than drawn from it.

Required DAC resolution (for N-bit ADC)	$N-2$
Number of needed C_u (for differential DAC)	$2 \cdot 2^N$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	lower one of $\{V_{in,n}, V_{in,p}\}$
Efficiency $(1 - E_{avg}/E_{avg,classic})$	96.87%
Remarks	Requires very accurate V_{cm} source

Table 3.7: Features of tri-level algorithm.

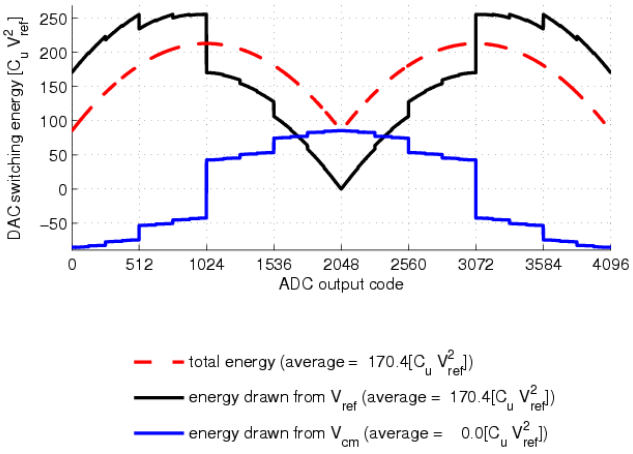


Figure 3.18: Energy consumption due to DAC switching for 12-bit ADC using tri-level algorithm.

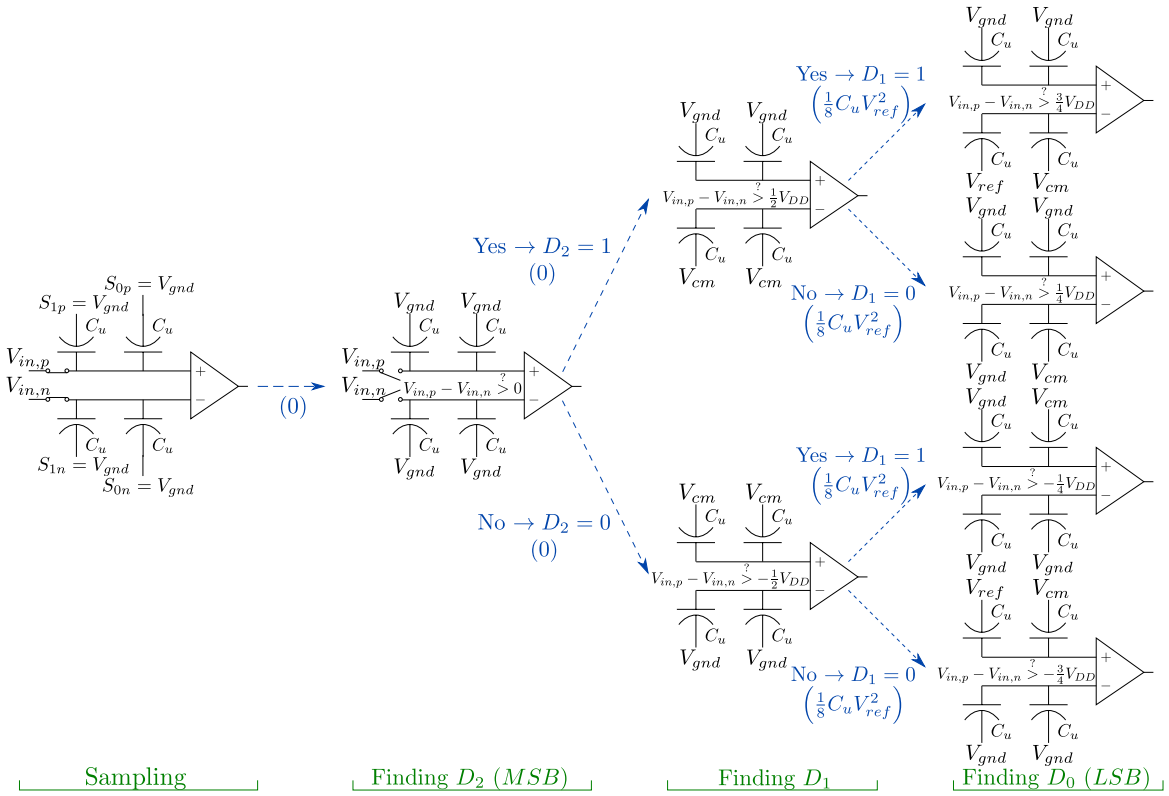


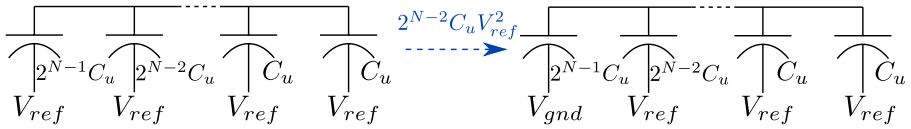
Figure 3.19: 3-bit SAR ADC incorporating tri-level algorithm.

3.1.8 Switchback algorithm

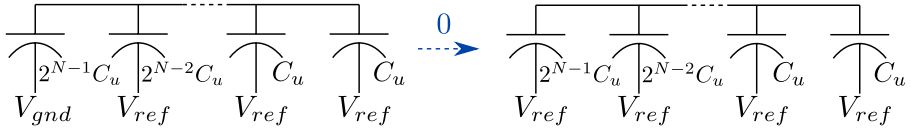
In switchback method (first described in [27], example of 3-bit ADC conversion stages is given in Figure 3.23) conversion starts with sampling the input on top plates of DACs while MSB capacitor's bottom plates are kept at V_{gnd} and bottom plates of all the rest of capacitors are held at V_{ref}. After sampling phase ends sampled voltages are compared and both value of D_{N-1} and MSB capacitors bottom plate voltages are decided:

- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, leading to $S_{N-2,p} \rightarrow V_{gnd}$ and $S_{N-2,n} \rightarrow V_{ref}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, leading to $S_{N-2,p} \rightarrow V_{ref}$ and $S_{N-2,n} \rightarrow V_{gnd}$

Such procedures enables switching of MSB capacitors without consuming energy. This improvement comes from observation that while energy from external source is needed when changing DAC's top-plate from V_{ref} to $\frac{1}{2}V_{ref}$ using MSB capacitor (changing DAC's input from $[11\cdots11]$ to $[01\cdots11]$, presented in Figure 3.20a, assuming that at start DAC is discharged), additional energy is not needed to discharge top plate from $\frac{1}{2}V_{ref}$ to V_{ref} using MSB capacitor (changing DAC's input from $[01\cdots11]$ to $[11\cdots11]$, presented in Figure 3.20b).



(a) Energy non-efficient sequence.



(b) Energy efficient sequence.

Figure 3.20: Ilustration of idea behind initial switching sequence

Calculation of energies shown in Figure 3.20 is done using equation 3.4:

- from $[11\cdots11]$ to $[01\cdots11]$ – Figure 3.20a:

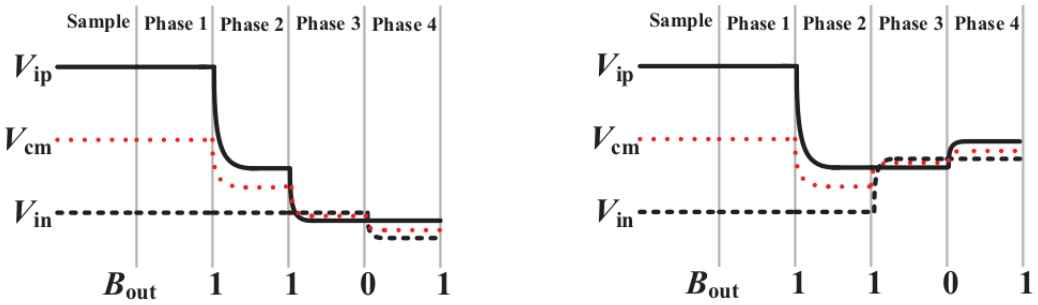
$$E = 2^{N-1}C_uV_{ref}\left[V_{ref} - \frac{2^{N-1}}{2^N}V_{ref} - (V_{ref} - V_{ref})\right] = 2^{N-2}C_uV_{ref}^2 \tag{3.5}$$

- from $[01\cdots11]$ to $[11\cdots11]$ – Figure 3.20b:

$$\begin{aligned} E =& 2^{N-1}C_uV_{ref}\left[V_{ref} - V_{ref} - \left(V_{ref} - \frac{2^{N-1}}{2^N}V_{ref}\right)\right] + \\ &+ 2^{N-1}C_uV_{ref}\left[V_{ref} - V_{ref} - \left(V_{gnd} - \frac{2^{N-1}}{2^N}V_{ref}\right)\right] = 0 \end{aligned} \tag{3.6}$$

During each subsequent transition only one capacitor is switched, but while first transition leads to increase of top plate potential of one of the DACs (by $\frac{1}{2}V_{ref}$), all other result in lowering

potential of appropriate DAC (by $\frac{1}{2^{i-N+2}}V_{ref}$ after resolving i-th bit as in other algorithms). This scheme allows for one-sided operation each cycle, as in monotonic switching, while maintaining convergence level of samples at V_{cm} (in worst case scenario initial common-mode level is $\frac{3}{4}V_{ref}$ and by the end of conversion is lowered to $\frac{1}{2}V_{ref}$). The comparison of DAC's top plate potentials for monotonic and switchback methods is presented on Figure 3.21.



(a) Monotonic algorithm **(b)** Switchback algorithm

Figure 3.21: Comparison of voltage on DACs top plates during conversion using monotonic and switchback algorithms. [27]

Although this method lowers required DAC resolution by 1-bit compared to classical method and is more power efficient, it requires DAC precharge before every conversion, which is very energy consuming (since DAC has a N-1 resolution, precharge energy can be calculated, based on equation 3.5, as $2 \cdot 2^{N-3}C_u V_{ref}^2$). Taking into account only power consumption during conversion the switchback algorithm has efficiency of 90.63% but a realistic calculation including precharge energy lowers this value to 71.87% (this difference can be seen on Figure 3.22). Additionally some time after each conversion is needed to perform this precharge, which extends time between subsequent conversions. Table 3.8 sums up basic features of switchback method.

Required DAC resolution (for N-bit ADC)	$N-1$
Number of needed C_u (for differential DAC)	$2 \cdot 2^N$
Required reference sources	V_{gnd}, V_{ref}
DAC's convergence voltage	worst case: $\frac{3}{4}V_{ref} \rightarrow V_{ref}$
Efficiency $(1 - E_{avg}/E_{avg,classic})$	71.87%
Remarks	Requires DAC precharge before every conversion

Table 3.8: Features of switchback algorithm.

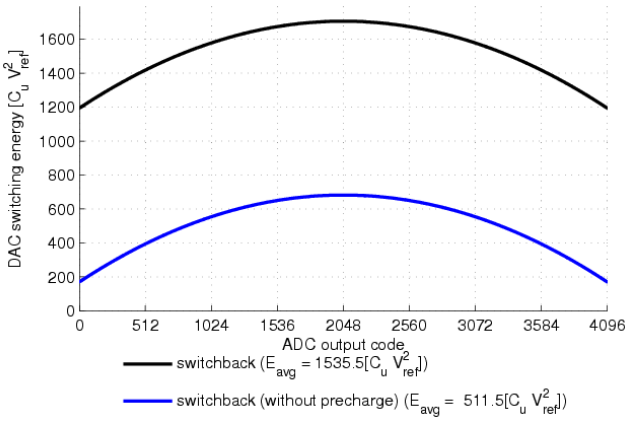


Figure 3.22: Energy consumption due to DAC switching for 12-bit ADC using switchback algorithm.

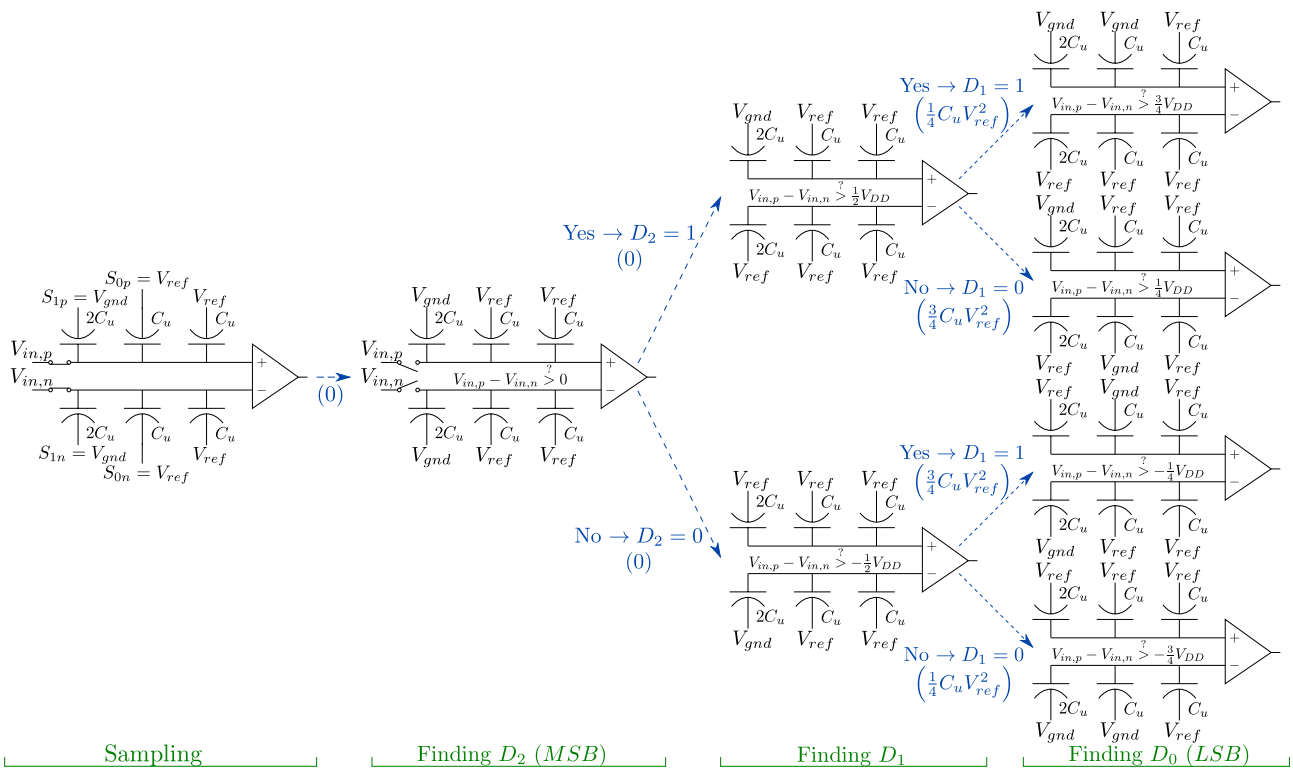
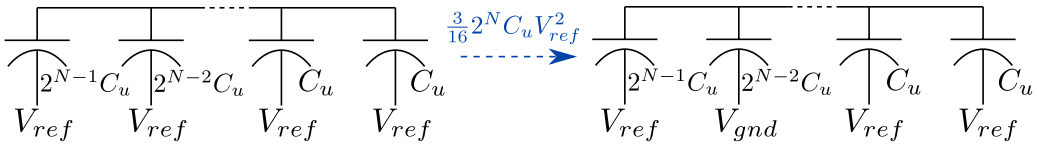


Figure 3.23: 3-bit SAR ADC incorporating switchback switching algorithm.

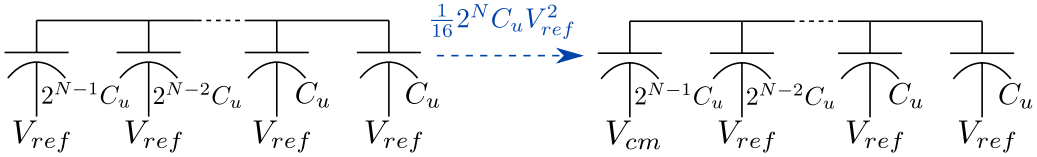
3.1.9 Improved switchback algorithm

Improved switchback algorithm [28] uses the same technique as switchback method to avoid power consumption when switching capacitors after resolving D_{N-1} (pre-charging both DACs to $\frac{1}{2}V_{ref}$ by making their inputs $[01\cdots11]$ and sampling on DACs top plates in such state). Also the same methodology of switching is applied – first switching increases voltage at one of the DACs top plate, while all further voltage changes lower appropriate DAC voltage level, so by the end of conversion process convergence of $V_{DAC,p}$ and $V_{DAC,n}$ to V_{cm} should be achieved. It also has the same disadvantage as switchback algorithm – it requires precharge before every conversion, which severely lowers power efficiency and adds more time between two conversions. Example of a 3-bit ADC following improved switchback algorithm is presented in Figure 3.25.

This algorithm incorporates an innovative technique to lower energy consumption – switching bottom plate of capacitor 2^iC_u ($i \neq \{0, N-1\}$) from V_{ref} to V_{gnd} will result in the same top plate voltage of the DAC as instead switching bottom plate of $2^{i+1}C_u$ from V_{ref} to V_{cm} but in latter case energetic cost will be lower (example comparing those two cases is presented at Figure 3.24).



(a) Energy non-efficient sequence.



(b) Energy efficient sequence.

Figure 3.24: Ilustration of idea behind energy-saving switching sequence.

Calculation of energies shown in Figure 3.24 is done using equation 3.4:

– $[11\cdots11]$ to $[10\cdots11]$ – Figure 3.24a:

$$E = \left(2^N - 2^{N-2}\right) C_u V_{ref} \left[V_{ref} - \frac{2^N - 2^{N-2}}{2^N} V_{ref} - (V_{ref} - V_{ref})\right] = \frac{3}{16} 2^N C_u V_{ref}^2 \quad (3.7)$$

– $[11\cdots11]$ to $[\frac{1}{2}1\cdots11]$ – Figure 3.24b:

$$\begin{aligned} E =& 2^{N-1} C_u V_{ref} \left[V_{ref} - \frac{2^N - 2^{N-2}}{2^N} V_{ref} - (V_{ref} - V_{ref})\right] + \\ &+ 2^{N-1} C_u V_{cm} \left[V_{cm} - \frac{2^N - 2^{N-2}}{2^N} V_{ref} - (V_{ref} - V_{ref})\right] \overset{V_{cm}=1/2 V_{ref}}{=} \frac{1}{16} 2^N C_u V_{ref}^2 \end{aligned} \quad (3.8)$$

To even further decrease number of needed unit capacitors the same technique as in AMCS described in 3.1.6 is used – two smallest capacitors in each DAC (connected to $S_{1p}, S_{0p}, S_{1n}, S_{0n}$ switches) have equal capacitance, rest is binary scaled. Switching algorithm proceeds as described above until D_1 is resolved – based on its value last DAC switching follows the rule:

- if $D_1 = 1$ than S_{0p}➛V_{cm}, S_{0n}➛V_{ref}
- if $D_1 = 0$ than S_{0p}➛V_{ref}, S_{0n}➛V_{cm}

After this switching a comparison resulting in deciding value of D_0 is carried out.

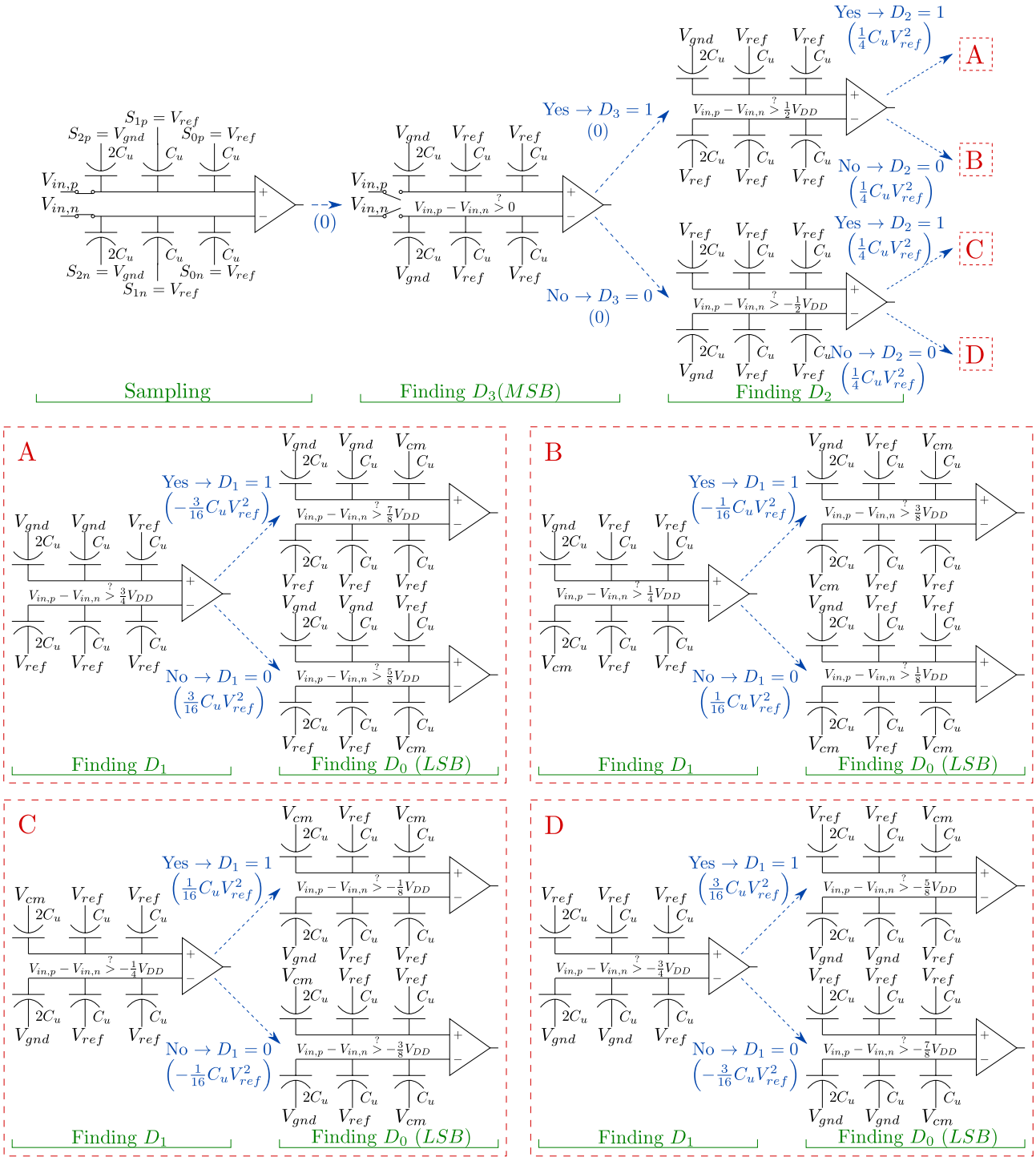


Figure 3.25: 4-bit SAR ADC incorporating Sanyal-Sun switching algorithm.

It is clear that improved switchback algorithm is the most complicated of all described successive approximation variants – it combines techniques used in other approaches to perform DAC switching during conversion in most efficient energy-wise way (in this case energy efficiency is 98.43%). This does not mean though, that it is the most energy efficient method overall – due to high complexity the digital logic will most likely require more power compared to other approaches and when a more realistic calculation including precharge energy is carried out, efficiency drops to 89.09% (precharge influence on power consumption is shown in Figure 3.27). Additional reference voltage source V_{cm} is required to provide a very accurate voltage level since it's actual voltage value influence differential DAC's output in all conversion phases. Power drawn from both reference sources for a 12-bit ADC is presented on Figure 3.26. Negative energy values on this figure mean that for given output code more energy is given back to source than drawn from it. Voltages on DACs top-plates have common-mode level behaving the same way as in switchback algorithm, the only difference is that at last conversion it is shifted by LSB due to asymmetric switching. Summary of features of improved switchback method is shown in Table 3.9.

Required DAC resolution (for N-bit ADC)	$N-2$
Number of needed C_u (for differential DAC)	$2 \cdot 2^N$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	worst case: $\frac{3}{4}V_{ref} \rightarrow V_{cm}-$LSB
Efficiency $(1 - E_{avg}/E_{avg,classic})$	82.17%
Remarks	Requires DAC precharge before every conversion

Table 3.9: Features of improved switchback algorithm.

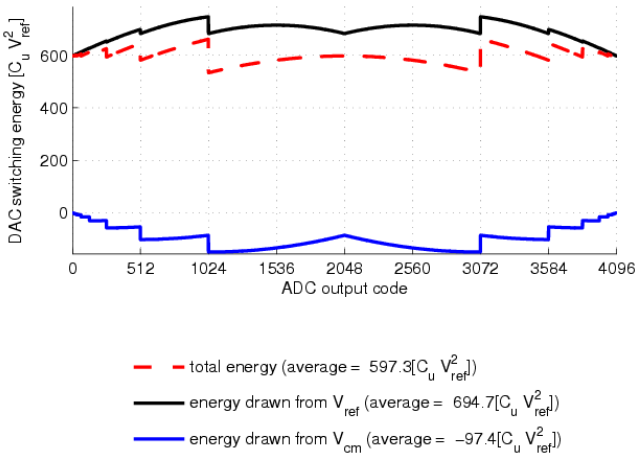


Figure 3.26: Energy consumption due to DAC switching for 12-bit ADC using improved switchback algorithm.

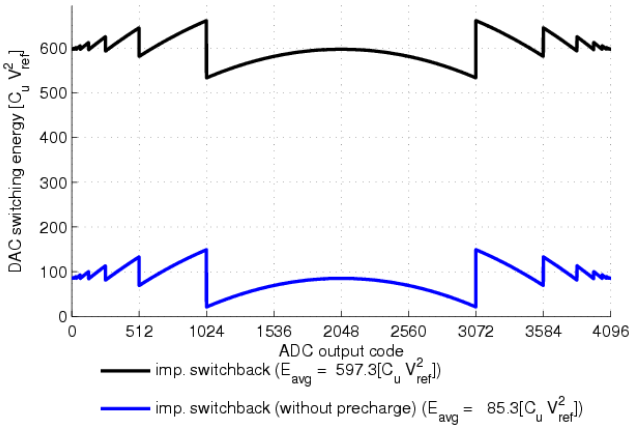


Figure 3.27: Comparison of power consumption due to DAC switching when including and excluding precharge for 12-bit ADC using improved switchback algorithm.

3.1.10 $V_{cm}-$based monotonic

The idea behind this variation of successive approximation is based upon using passive voltage shift (voltage level change without any charge change on DAC) to reduce energy needed during DAC switching [29], as presented in Figure 3.30.

Conversion starts with sampling input on top plates of capacitive DACs, while all bottom-plates are kept at V_{cm}. When sampling ends top plates are disconnected from input, bottom plates are kept at V_{cm} and first comparison is performed. Based on the result value of D_{N-1} and next step of approximation are decided:

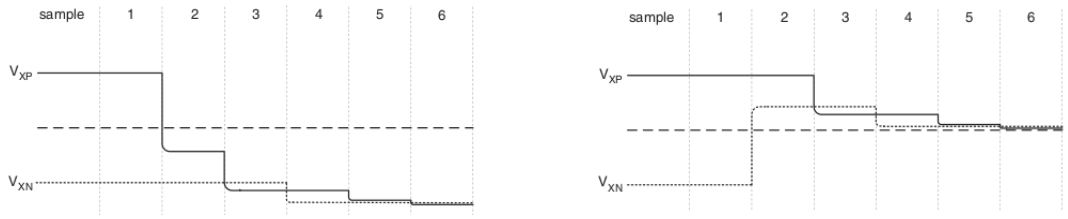
- $D_{N-1} = 1$ if $V_{in,p} > V_{in,n}$, resulting in $\forall_{i=0}^{2^{N-2}} S_{ip} \rightarrow V_{cm}, S_{in} \rightarrow V_{ref}$
- $D_{N-1} = 0$ if $V_{in,p} < V_{in,n}$, resulting in $\forall_{i=0}^{2^{N-2}} S_{ip} \rightarrow V_{ref}, S_{in} \rightarrow V_{cm}$

This means that after first bit is decided all switches of one DAC remain unaffected, while all switches on other DAC are set to V_{ref} resulting in mentioned passive voltage shift, and overall no energy drawn from sources in this step. All further DAC switching is performed following rules stated in Table 3.10.

Table 3.10: Reference voltages choice after deciding value of D_{N-1} in V_{cm}-based monotonic switching.

	$D_{N-1} = 1$		$D_{N-1} = 0$	
DAC side	$V_{in,p}$	$V_{in,n}$	$V_{in,p}$	$V_{in,n}$
S_{i-1} if $D_i = 1$	V_{gnd}	V_{ref}	V_{cm}	V_{cm}
S_{i-1} if $D_i = 0$	V_{cm}	V_{cm}	V_{ref}	V_{gnd}

This switching procedure results in variable DAC's convergence and common-mode levels as shown in Figure 3.28 , which leads to a need of comparator capable to work with quite wide range of common-mode voltages.



(a) Monotonic algorithm

(b) V_{cm}-based monotonic algorithm

Figure 3.28: Comparison of voltage on DACs top plates during conversion using monotonic and V_{cm}-based monotonic algorithms. [29]

This method of conversion is the most energy efficient among all described in this chapter (average switching energy is 97.65% lower than in classical method, as can be noticed from comparison of Figure 3.29 and 3.3) and requires 2-bit lower DAC resolution compared to classical method. Disadvantage of this method is a requirement to supply additional reference voltage source V_{cm} capable of providing a very accurate voltage level since it's actual voltage value influence differential DAC's output in all conversion phases. Power drawn from both reference sources for a 12-bit ADC is presented on Figure 3.29. Negative energy values on this figure mean that for given output code more energy is given back to source than drawn from it.

Required DAC resolution (for N-bit ADC)	$N-2$
Number of needed C_u (for differential DAC)	$2 \cdot 2^N$
Required reference sources	V_{gnd}, V_{cm}, V_{ref}
DAC's convergence voltage	worst case: $\frac{3}{4}V_{ref} \rightarrow V_{cm}$
Efficiency $(1 - E_{avg}/E_{avg,classic})$	97.65%
Remarks	Requires very accurate V_{cm} source

Table 3.11: Features of V_{cm}-based monotonic algorithm.

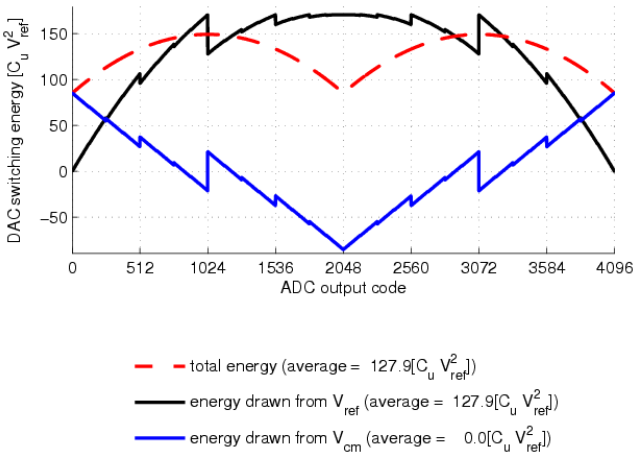


Figure 3.29: Energy consumption due to DAC switching for 12-bit ADC using V_{cm}-based monotonic algorithm.

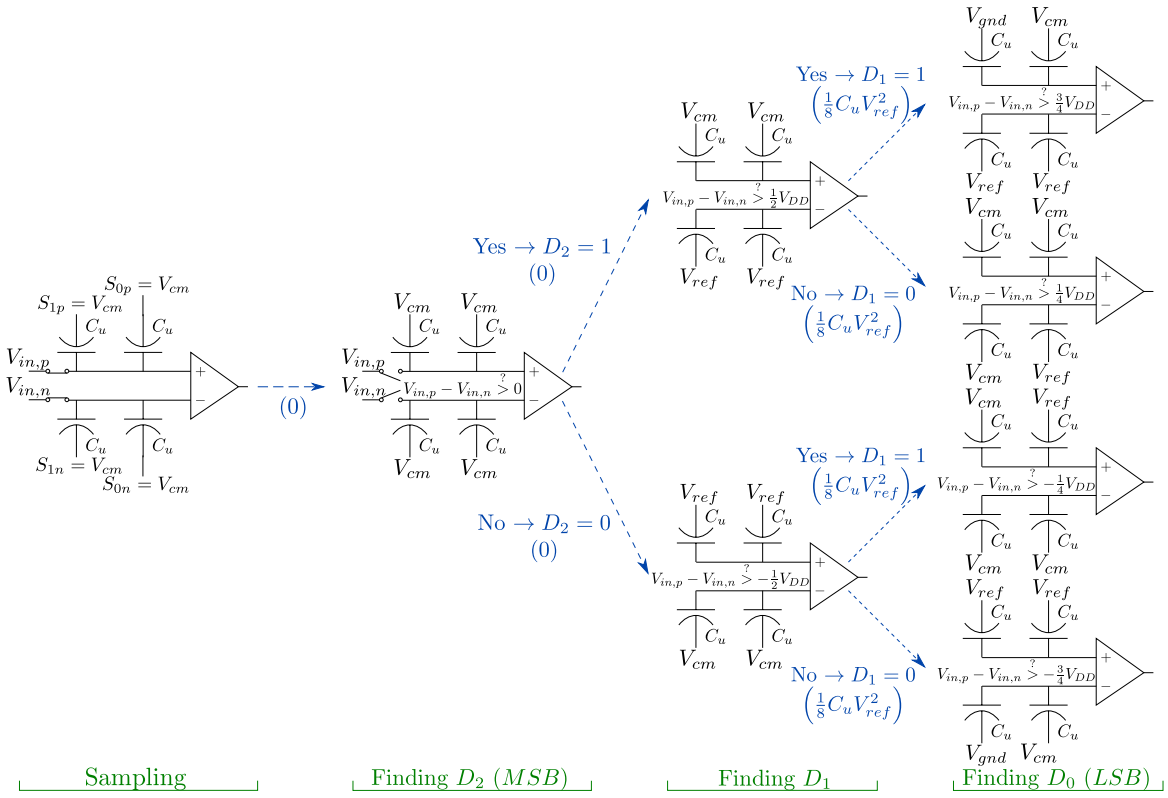


Figure 3.30: 3-bit SAR ADC incorporating V_{cm} algorithm.

3.1.11 Comparison of power consumption of presented SAR algorithms

Most important characteristics of all described methods are presented in Table 3.12, while Figure 3.31 shows comparison of power consumption due to DAC switching for 12-bit ADCs incorporating different SAR algorithm variants (on X-axis output code of ADC is marked, while Y-axis presents energy consumption in technology-independent unit $C_UV_{ref}^2$). It can be clearly seen that thanks to new approaches to SAR algorithm the energy consumption has been lowered, but it must be noted that this figure presents only DAC switching energy consumption and consumption of other blocks is omitted – this observation might cause some approaches to loose their attractiveness due to high power consumption of e.g. more complicated logic. Energy consumption grows very quickly with increasing ADC's resolution (as presented on Figure 3.32), but all described algorithms keep their energy efficiency in shown resolution range.

Variable convergence level of sampled voltages in some SAR variants (monotonic, tri-level) also causes difficulties that cannot be seen when just comparing algorithms based on Figure 3.31. For those two reasons Merged Capacitor Switching algorithm has been chosen to be implemented in the presented work – constant convergence level combined with relatively uncomplicated switching sequence (simple digital logic) and usage of top plate sampling appeared to be a good starting point for a very low-power SAR ADC design. Although EMCS algorithm appears to have very similar characteristics with added bonus of improving linearity, it has

also the disadvantages of more complex digital logic and a need of two-phase DAC switching (leading to longer time needed for conversion).

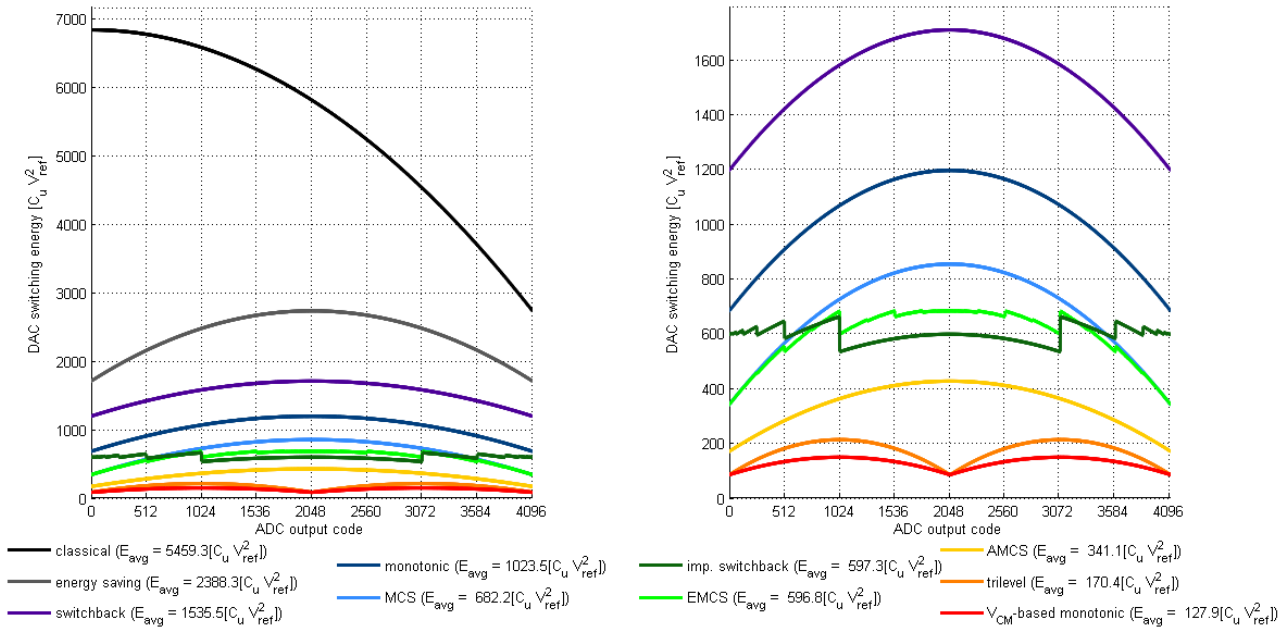


Figure 3.31: Comparison of power consumption due to DAC switching for different SAR algorithm variants.

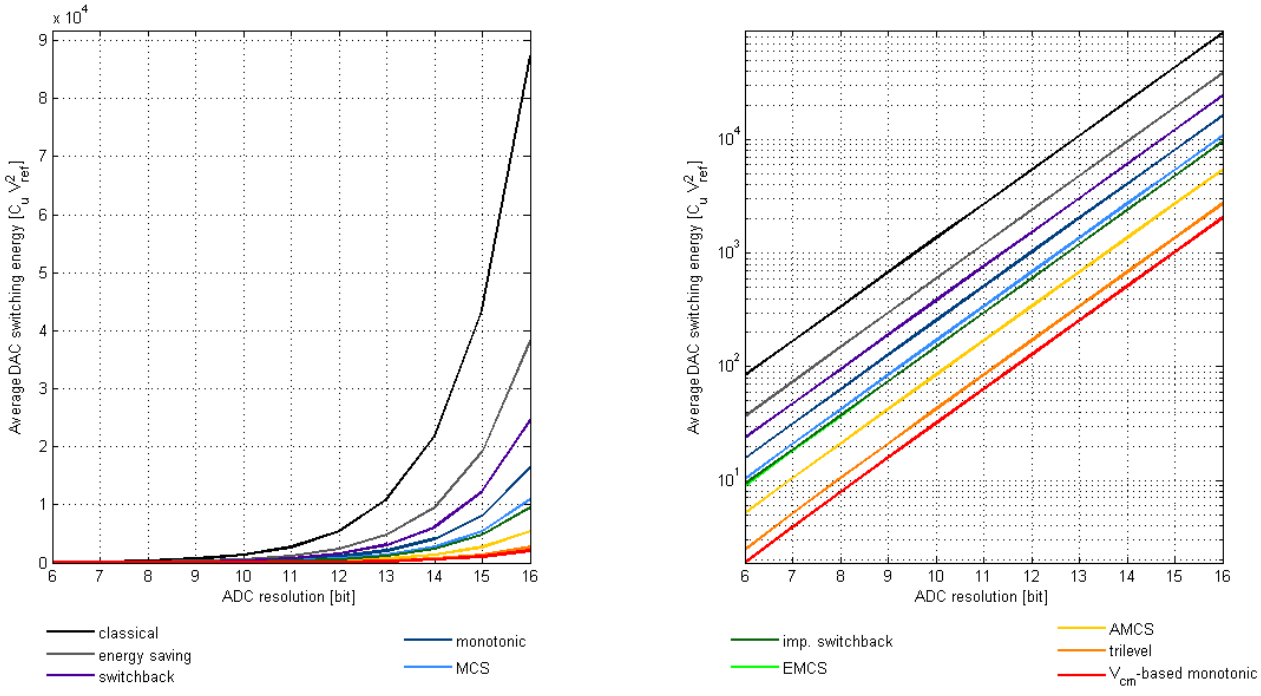


Figure 3.32: Comparison of average DACs' switching energy for different resolutions. For ease of comparison two versions of plot are presented – with linear Y-axis scale (left) and logarithmic Y-axis scale (right).

Table 3.12: Comparison of average switching energy for different algorithms and different ADC's number of bits N.

Algorithm	Needed DAC resolution (for N-bit ADC)	Number of needed C_u (for diff. DAC)	$E_{avg}[C_u V_{ref}^2]$ (N=12)	Efficiency	Needed reference sources	DAC's convergence voltage
classical	N	$2 \cdot 2^N$	5459.5	reference (0%)	$V_{gnd}, V_{cm}^1, V_{ref}$	$\frac{1}{2}V_{ref}$
energy saving	N	$2 \cdot 2^N$	2388.3	56.25%	V_{gnd}, V_{ref}	$\frac{1}{2}V_{ref}$
monotonic	N-1	$2 \cdot 2^{N-1}$	1023.5	81.25%	V_{gnd}, V_{ref}	$\frac{1}{2}V_{ref} \rightarrow V_{gnd}$
MCS	N-1	$2 \cdot 2^{N-1}$	682.2	87.5%	$V_{gnd}, V_{cm}^1, V_{ref}$	V_{cm}
EMCS	N-1	$2 \cdot 2^{N-1}$	596.8	89.07%	$V_{gnd}, V_{cm}^1, V_{ref}$	V_{cm}
AMCS	N-2	$2 \cdot 2^{N-2}$	341.1	93.75%	$V_{gnd}, V_{cm}^2, V_{ref}$	$V_{cm} - LSB$
trilevel	N-2	$2 \cdot 2^{N-2}$	170.4	96.87%	$V_{gnd}, V_{cm}^3, V_{ref}$	lower one of $\{V_{in,n}, V_{in,p}\}$
switchback	N-1	$2 \cdot 2^{N-1}$	1535.5	71.87%	V_{gnd}, V_{ref}	worst case: $\frac{3}{4}V_{ref} \rightarrow V_{ref}$
improved switchback	N-2	$2 \cdot 2^{N-2}$	597.3	82.17%	$V_{gnd}, V_{cm}^3, V_{ref}$	worst case: $\frac{3}{4}V_{ref} \rightarrow V_{cm} - LSB$
V_{cm}-based monotonic	N-2	$2 \cdot 2^{N-2}$	127.9	97.6%	$V_{gnd}, V_{cm}^3, V_{ref}$	worst case: $\frac{3}{4}V_{ref} \rightarrow V_{cm}$

[1] – value of supplied voltage does not need to be very accurate, since it have no influence on result of conversion

[2] – value of supplied voltage should be accurate, since it have a direct influence on resolving D_0

[3] – value of supplied voltage must be very accurate, since it is relied upon in all conversion phases

3.2 Digital-to-analog converter

Digital-to-analog converter is used in SAR ADC as reference voltage level generator controlled by SAR logic. Since presented design incorporates MCS algorithm a 11-bit DAC is needed (top plate sampling allows for first comparison without any DAC switching), which would require a simple binary-weighted charge scaling DAC build out of 2048 unit capacitances (as can be can seen from Figure 3.1). Such large number of capacitors would undoubtedly lead to very big layout area and high total capacitance (unless a very small capacitor having good matching properties would be available, which is not the case in IBM CMRF8SF technology).

To remedy this a split binary-weighted architecture was used in this design.

3.2.1 Split binary-weighted DAC

A split binary-weighted DAC, shown in Figure 3.33 consists of two binary-weighted arrays: one M-bit and one L-bit, where $M+L=N$. Those arrays are connected with bridge capacitor C_B which size should be chosen in such a way that its capacitance in series with L-bit DAC capacitance (when all capacitors' bottom plate switches in L-bit DAC are connected to V_{gnd}) is equivalent to C_u [12]:

$$\frac{C_B \cdot 2^L C_u}{C_B + 2^L C_u} = C_u \Rightarrow C_B = \frac{2^L}{2^L - 1} C_u \tag{3.9}$$

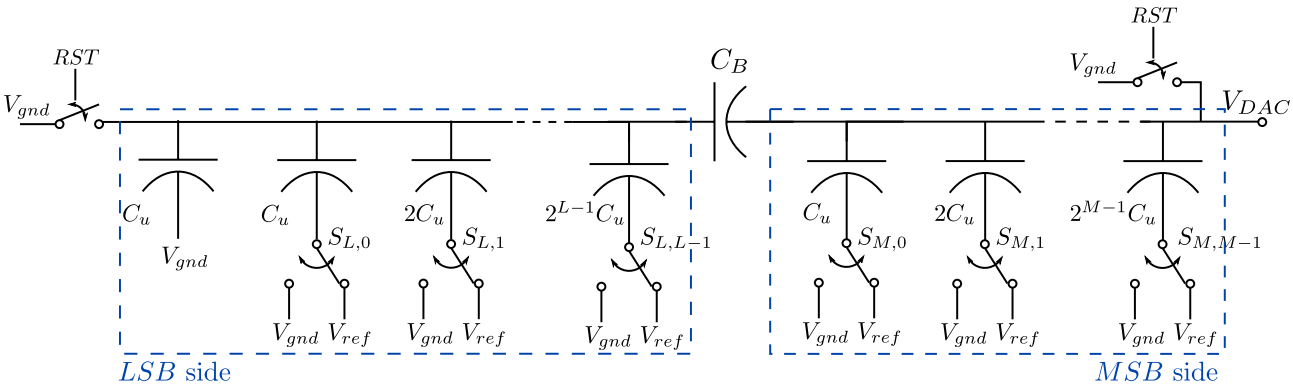


Figure 3.33: Schematic of charge scaling split binary-weighted N-bit DAC.

Gain error

Value of C_B obtained through equation 3.9 is a fraction of C_u – implementation of such value would lead to mismatch between bridge capacitor and the rest of array and possibly lead to difficulties in layout. For those reasons often instead of using ideal value of C_B a single unit capacitance C_u is used [9]. This modification is a cause of gain error in conversion. Additionally to save area the dummy capacitor (always connected to V_{gnd}) in L-bit DAC can be removed – its role is only to provide precisely binary voltage division, so its removal will also cause only constant gain error. To see how output voltage changes due to those errors first calculation for ideal case (ideal value of C_B and additional dummy C_u in L-bit DAC) will be done (calculations will be carried out for final state of DAC i.e. all capacitors will be connected to either V_{gnd} or V_{ref}).

First case to be considered is situation when entire L-bit DAC is connected to V_{gnd} and number of capacitors from M-bit DAC are connected to V_{ref} (presented in Figure 3.34). Voltage V_{DAC} can be expressed as ($C^M_{V_{ref}}$ denotes total capacitance connected to V_{ref}, while effective

capacitance of series connection of L-bit DAC and C_B is denoted C_{eL}):

$$V_{DAC} = \frac{C^M_{V_{ref}}}{(2^M-1)C_u + C_{eL}} V_{ref} = \left\| C_{eL} = C_u \right\| = \frac{C^M_{V_{ref}}}{2^M C_u} V_{ref} \tag{3.10}$$

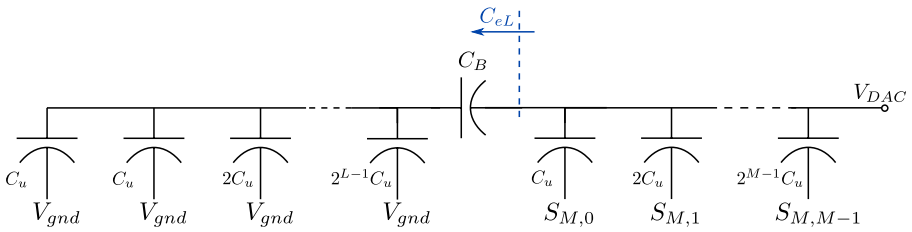


Figure 3.34: Schematic of split DAC with L-bit DAC connected to V_{gnd}.

Second case to be considered is situation when entire M-bit DAC is connected to V_{gnd} and number of capacitors from L-bit DAC are connected to V_{ref} (presented in Figure 3.35). Voltage V_L can be expressed as (where $C^L_{V_{ref}}$ is total capacitance connected to V_{ref}, while effective capacitance of series connection of M-bit DAC and C_B is denoted C_{eM}):

$$V_L = \frac{C^L_{V_{ref}}}{2^L C_u + C_{eM}} V_{ref} = \left\| C_{eM} = \frac{C_B \cdot (2^M-1)C_u}{C_B + (2^M-1)C_u} \right\| = \frac{C^L_{V_{ref}}[2^L + (2^M-1)(2^L-1)]}{2^L C_u[2^L + (2^M-1)(2^L-1) + 2^M - 1]} V_{ref} \tag{3.11}$$

Leading to:

$$V_{DAC} = \frac{C_B}{(2^M-1)C_u + C_B} V_L = \frac{2^L}{(2^M-1)(2^L-1) + 2^L} V_L = \frac{C^L_{V_{ref}}}{2^N C_u} V_{ref} \tag{3.12}$$

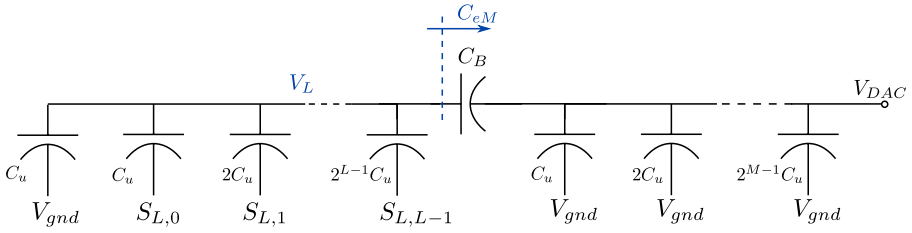


Figure 3.35: Schematic of split DAC with M-bit DAC connected to V_{gnd}.

Combining equation 3.10, 3.12 and using superposition principle DAC output voltage is obtained:

$$V_{DAC,ideal} = \frac{2^L C^M_{V_{ref}} + C^L_{V_{ref}}}{2^N C_u} V_{ref} \tag{3.13}$$

Following the same methodology for situation where $C_B = C_u$ and dummy capacitor in

L-bit DAC is removed results in:

$$V_{DAC,M-DAC\rightarrow V_{gnd}} = \frac{C^M_{V_{ref}}}{2^M(1-2^{-N})C_u}V_{ref} \tag{3.14}$$

$$V_{DAC,L-DAC\rightarrow V_{gnd}} = \frac{1}{2^M}\cdot\frac{C^L_{V_{ref}}}{2^L(1-2^{-N})C_u}V_{ref} \tag{3.15}$$

$$V_{DAC,non-ideal} = \frac{1}{1-2^{-N}}\cdot\frac{2^LC^M_{V_{ref}}+C^L_{V_{ref}}}{2^NC_u}V_{ref} \tag{3.16}$$

Comparison of equations 3.13 and 3.16 reveals that gain error introduced by non-fractional value of C_B and removal of dummy capacitor is equal to $\frac{1}{1-2^{-N}}$. This small constant value (for given resolution) can be calibrated digitally if needed.

Value of unit capacitance C_u

Capacitance of unit capacitor C_u should be kept as small as possible to save power and area. On the other hand using too small value will lead to high mismatch and noise influence. Value of capacitance that would allow for reliable operation of DAC in respect to mismatch and thermal noise will be calculated in two next paragraphs.

– **Mismatch limited capacitance**

Unit capacitor can be characterised using its value C_u and standard deviation σ_u. Those values can be correlated with their layout parameters by [30]:

$$C_u = K_C\cdot A_u \tag{3.17}$$

$$\frac{\sigma_u}{C_u} = \frac{1}{\sqrt{2}}\cdot\frac{K_\sigma}{A_u} \tag{3.18}$$

where A_u is area of unit capacitor, K_C is capacitor density parameter and K_σ is technology dependant matching parameter.

To calculate value assuring safe operation of classical DAC (like the one shown in Figure 3.1) a worst-case deviation of non-linearity is selected as starting-point – for this DAC such value is differential non-linearity at MSB code transition expressed as [30]:

$$\sigma_{DNL} = \sqrt{2^N-1}\frac{\sigma_u}{C_u}\cdot\text{LSB} \tag{3.19}$$

Assuming that $3\sigma_{DNL} < 1LSB$ (to achieve high reliability) and combining equations 3.17, 3.18 and 3.19 results in mismatch limited value of unit capacitance for classical DAC $C_{u,classic}$(for differential configuration, which requires $\sqrt{2}$ lower unit capacitance since voltage range is 2 times bigger and mismatch error rises only $\sqrt{2}$ times):

$$C_u = \frac{9}{2\sqrt{2}}\left(2^N-1\right)K_\sigma^2K_C \tag{3.20}$$

As an example unit capacitor value for MIM-capacitor in IMB130nm CMRF8SF technology $\left(K_\sigma = 4.12\frac{\%}{\mu m}, K_C = 2.05\mu m\right)$ for 11-bit DAC should have capacitance $C_{u,MIM,classical}$.

$$C_{u,MIM,classical} = 22.7fF \tag{3.21}$$

Based on equations 3.14 and 3.15 (which show that L-side DAC influence on output voltage is 2^L smaller than that of M-side DAC) one can assume that for for $M \geq \frac{N}{2}$ the M-side of DAC will be the main contributor to mismatch induced error. From equation 3.19 a worst-case standard deviation of DNL for M-bit DAC can be calculated as:

$$\sigma_{DNL,M} = \sqrt{2^M - 1}\frac{\sigma_u}{C_u} \cdot \frac{V_{ref}}{2^M} \tag{3.22}$$

Combining equation above with equations 3.17, 3.18 and assuming that $3\sigma_{DNL,M} < 1 \cdot \text{LSB}$ leads to mismatch limited value of unit capacitance for differential split DAC $C_{u,split}$ defined as:

$$C_{u,split} = \frac{9}{2\sqrt{2}}2^{2(N-M)}\left(2^M - 1\right)K_\sigma^2 K_C \tag{3.23}$$

Using the same capacitor as in previous example, unit capacitor for split DAC with M=7, N=4 has capacitance $C_{u,MIM,split}$.

$$C_{u,MIM,classical} = 359.89fF \tag{3.24}$$

– **Thermal noise limited capacitance**

Thermal noise generate by capacitive DAC with total capacitance C_{tot} is equal to $\overline{V_{therm}^2}$:

$$\overline{V_{therm}^2} = \frac{k_B T}{C_{tot}} \tag{3.25}$$

where $k_B = 1.38 \cdot 10^{-23}\left[\frac{J}{K}\right]$ is Boltzmann constant and T is temperature. Combining equations 2.1, 3.25 and assuming similarly to previous case that $3\sqrt{\overline{V_{therm}^2}} < 1\text{LSB}$ thermal limited total DAC capacitance $C_{tot,therm}$:

$$C_{tot,therm} = 9\frac{2^{2N}}{V_{ref}^2}k_B T \tag{3.26}$$

Applying this equation to considered case ($N = 11, T = 293K, V_{ref} = 1.2V$) yields:

$$C_{tot,therm} \approx 105.3[fF] \tag{3.27}$$

Comparing values of $C_{u,MIM,classical}, C_{u,MIM,split}$ and $C_{tot,therm}$ (respectively equations 3.21, 3.24 and 3.27) it becomes apparent that influence of thermal noise is negligible. The other thing worth noting is the difference in capacitance of $C_{u,MIM,classical}$ and $C_{u,MIM,split}$ – their values indicate that while split DAC uses many times less unit capacitors, for some configu-

rations it might turn out to have similar total capacitance to classical DAC due to mismatch limitations. In such cases choice between classical and split DAC might be dictated by minimal unit capacitance allowable by DRC rules (e.g. in IMB130nm CMRF8SF minimal capacitance of MIM-capacitor is 60fF, meaning that $C_{u,MIM,classical}$ is impossible to implement).

3.2.2 Split DACs comparison

Split DAC architecture allows for substantial reduction in number of unit capacitors needed to construct 11-bit DAC – Table 3.13 presents comparison of few possible configurations (output capacitance of DAC (implementation of $C_B = C_u$ is assumed) is denoted C_{out}, while value of mismatch limited unit capacitance is presented in technology-independent form $\frac{C_u}{K_\sigma^2 K_C}$, where C_u is calculated based on equation 3.23). Another benefit of split DAC approach is lowering energy needed to perform DAC switching (when compared to classical DAC with the same resolution and using the same size of C_u) – Figure 3.36 presents energy consumption for different configurations of 10-bit split DAC (using classical algorithm).

Table 3.13: Comparison of different configurations of 11-bit DAC. Values are shown for single DAC.

M-bit	L-bit	Number of C_u in DAC	$C_{out}[C_u]$	$\frac{C_u}{K_\sigma^2 K_C}$
5	6	95	31.98	404305
6	5	95	63.97	205275
7	4	143	137.94	103452
8	3	263	255.87	51929
9	2	515	511.75	26015
10	1	1025	1023.5	13020
11 (without split)	0	2047	2047	6513

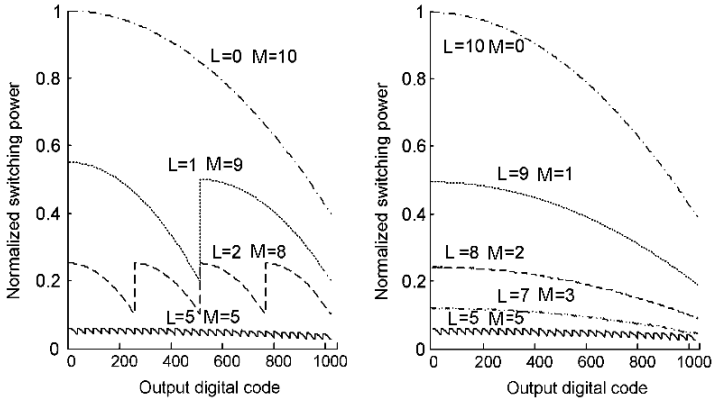


Figure 3.36: Normalized switching power for different distributions of L and M bit values for 10-bit DAC using classical switching algorithm. [32]

It can be seen from Figure 3.36 that more equal the distribution of L and M is, more power efficient DAC will be. The trade-off is higher value of unit capacitance and potentially greater influence of capacitance C_B on linearity of converter.

3.3 Sampling switch

To sample input signal onto DAC a device that connects and disconnect DAC from input is needed – a sampling switch. In simplest implementation (shown in Figure 3.37) such circuit can be just a single transistor (nMOS or pMOS) which through control of gate potential is turned on and off by V_{CLK} when needed.

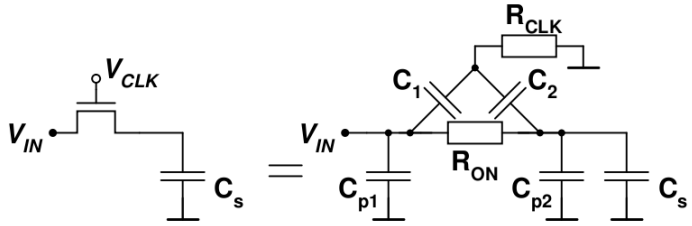


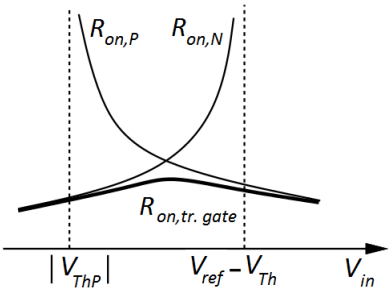
Figure 3.37: Single nMOS sampling switch and its RC equivalent. [34]

Figure 3.37 presents also an RC equivalent of this single transistor structure, which shows that sampling can be thought of as charging and discharging capacitor through resistor. Here a problem arises – while load capacitance has a rather constant value (in respect to input voltage), resistance of turned on transistor R_{on} will change depending on level of input signal as presented in Figure 3.38a. If a set time is allocated for sampling, such variable resistance and in turn variable value of RC would result in variable sampling accuracy of input signal. There are two ways to remedy this situation:

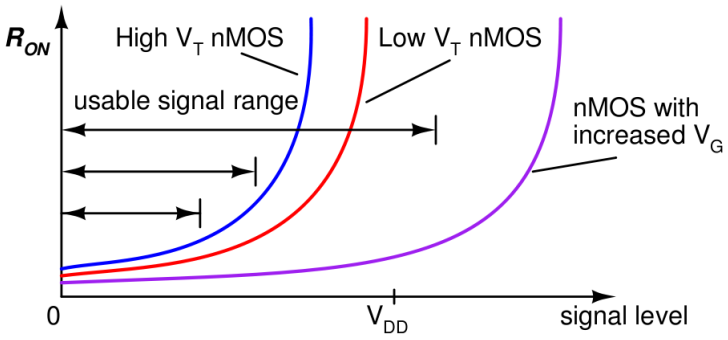
– circuit solution – using switch architectures that allows for lower transistor on-resistance e.g. transmission gate (resistance $R_{on,tr.\ gate}$ presented on Figure 3.38a) or bootstrapped

switch (resistance presented on Figure 3.38b – nMOS with increased V_G)

– technological solution – using modified, non-standard transistors e.g. with lowered threshold voltage (comparison between different transistor;s on-resistance is presented in Figure 3.38b)



(a) R_{on} for nMOS, pMOS and transmission gate ($R_{on,eq}$) [35]



(b) R_{on} of different nMOS switches [34]

Figure 3.38: On-resistance R_{on} of transistor switches as a function of input signal level.

Comparing Figures 3.38a and 3.38b it can be clearly seen that while transmission gate has much lower R_{on} over whole signal range compared to simple nMOS switch, there still is dependence between input signal level and switch on-resistance. For that reason a more complex bootstrapped switch seems to be a more reliable solution.

On-resistance can be approximated with equation (valid only for $V_{GS} \geq V_{Th}$, when $V_{GS} < V_{Th}$ switch is turned-off and its resistance becomes very high, ideally infinite) [34]:

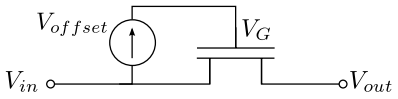
$$R_{on} \approx \frac{L}{\mu_0 C_{ox} W \left(V_{GS} - V_{Th}\right)} \tag{3.28}$$

where W is transistor's width, L is its length, μ_0 is charge carrier effective mobility and C_{ox} is gate oxide capacitance per area. From equation 3.28 the idea behind bootstrapped switch can be seen – if V_{GS} was to be kept constant and high regardless of input's signal voltage, than R_{on} would always have constant, low value. This can be realised, on conceptual level, by adding a voltage source V_{offset} between transistor's source and gate (shown in Figure 3.39a) of such value that:

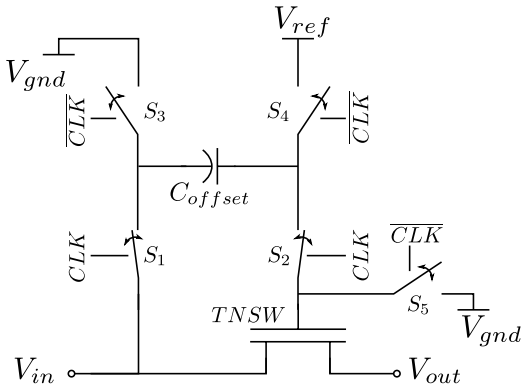
$$V_{GS} = V_{offset} = V_{ref} \tag{3.29}$$

Circuit that would act in that fashion is presented on Figure 3.39b. It consists of an nMOS transistor $TNSW$, capacitor C_{offset} and five switches $S_1 \div S_5$ ($[S_1, S_2, S_5]$ are controlled in complementary way to $[S_3, S_4]$). When CLK is low C_{offset} is charged to V_{ref} while transistor's gate is kept at V_{gnd} (circuit is turned off). When CLK goes high capacitor is disconnected from voltage source and connected between transistor's source and gate (S_5 is open now, so transistor's gate is no more connected to V_{gnd}) – such modes of operation effectively realise

circuit from Figure 3.39a.



(a) Concept of bootstrapped switch.



(b) Basic bootstrapped switch.

Figure 3.39: Bootstrapped switch concept.

Charge injection

When a MOS transistor is conducting, a finite amount of charge carriers are present in its channel. When transistor is turned off those carriers (and associated with them electrical charge) are distributed among source and drain. Total charge that is distributed in this process Q_{ch} can be calculated as [34, 35]:

$$Q_{ch} = WLC_{ox}\left(V_{Gon} - V_{Th}\right) \tag{3.30}$$

where V_{Gon} is transistor's gate voltage in on-state. This injection of charge results in distortion of sampled voltage ΔV_{ch} (figurative example shown on Figure 3.40) which can be expressed as:

$$\Delta V_{ch} = \alpha_Q \frac{Q_{ch}}{C_2} \tag{3.31}$$

where α_Q is a fraction of Q_{ch} that was injected to C_2.

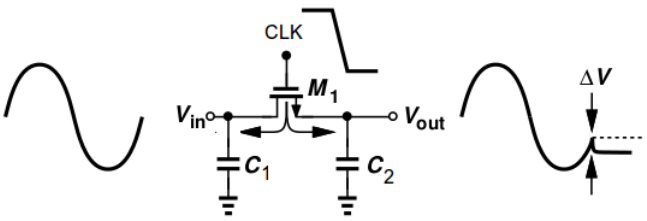


Figure 3.40: Charge injection when turning switch off and its influence on sampled voltage. [35]

Function describing charge distribution among transistor's source and drain is quite complex and depends on many parameters (e.g. impedance seen from each transistor's terminal, slope of CLK signal controlling gate voltage) but numerical solution can be found [34], based on which plot shown in Figure 3.41 can be obtained. Curves seen on this figure represent different values

of $\frac{C_1}{C_2}$, parameter on X-axis is a defined as $B = (V_{Gon} - V_{Th})\sqrt{\frac{\frac{W}{L}\mu_0 C_{ox}}{aC_2}}$, where a is a slope of CLK signal. Examination of Figure 3.41 leads to a conclusion that distribution of charge from charge injection effect depends heavily on ratio of C_1 to C_2. For SAR ADC this means that without knowledge about input signal driver's parameters no calculations of charge injected into ADC can be made, therefore it is very hard to implement any precautions measures.

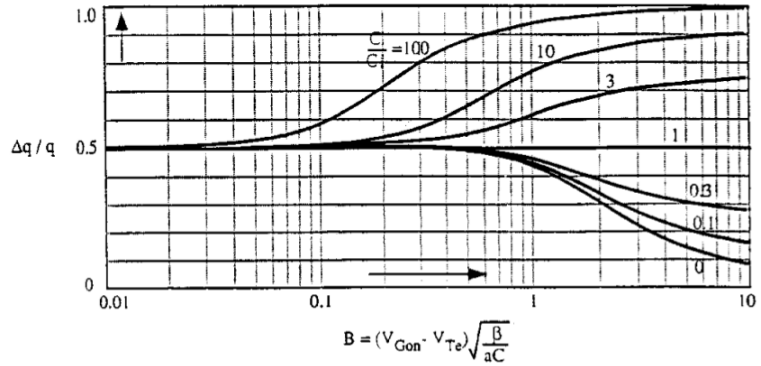


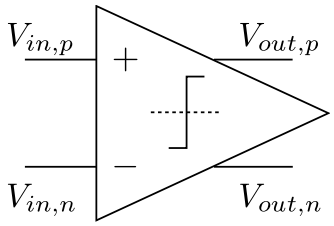
Figure 3.41: Charge injection distribution among transistor's source and drain. Reproduced from [34].

3.4 Comparator

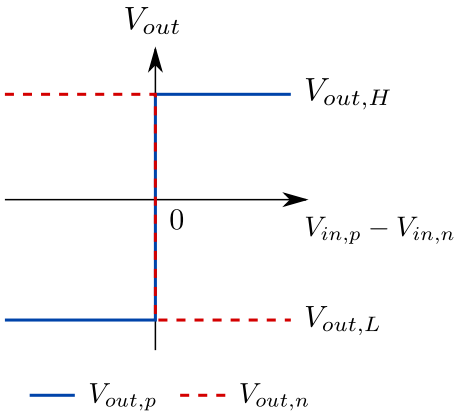
Function of a comparator is to compare two analog input signals (or one signal and a reference source voltage) and decide which of them has higher voltage level. A differential comparator has two inputs and two outputs (as presented in Figure 3.42a) and in an ideal case exhibits a transfer curve as the one in Figure 3.42b, which means that:

- if $V_{in,p} - V_{in,n} > 0$ than $V_{out,p} = V_{out,H}$ and $V_{out,n} = V_{out,L}$
- if $V_{in,p} - V_{in,n} < 0$ than $V_{out,p} = V_{out,L}$ and $V_{out,n} = V_{out,H}$

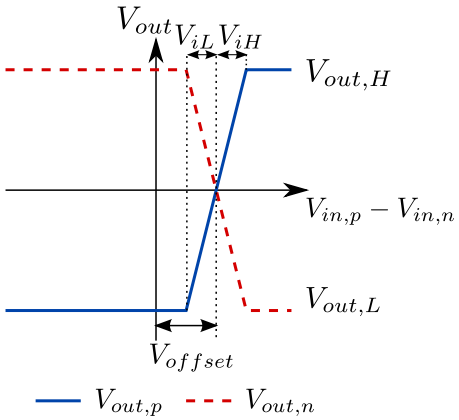
Output voltage level $V_{out,H}, V_{out,L}$ are in vast majority of implementations set to V_{ref}, V_{gnd} respectively.



(a) Symbol of differential comparator.



(b) Transfer curve of ideal differential comparator.



(c) Transfer curve of realistic differential comparator.

Figure 3.42: Differential comparator symbol and its transfer curve (ideal and realistic).

When effects of unideal behaviour of circuit (e.g. offset voltage V_{offset} or finite gain) are taken into account the transfer curve changes – an example of a more realistic transfer curve is presented in Figure 3.42c. Behaviour of comparator changes in such case to:

- if $V_{in,p} - V_{in,n} > V_{iH}$ than $V_{out,p} = V_{out,H}$ and $V_{out,n} = V_{out,L}$
- if $V_{in,p} - V_{in,n} < V_{offset} - V_{iL}$ than $V_{out,p} = V_{out,L}$ and $V_{out,n} = V_{out,H}$

Comparators used in CMOS technology can be divided into three general groups:

- open-loop comparators (example shown in Figure 3.43) – operational amplifiers without frequency compensation acting as continuous time comparators. Lack of compensation does not cause problems in this case since precise value of gain and its linearity are not necessary, but due to limited gain-bandwidth product this kind of comparators are rather slow in relation to other architectures. Additional disadvantage is static power consumption.

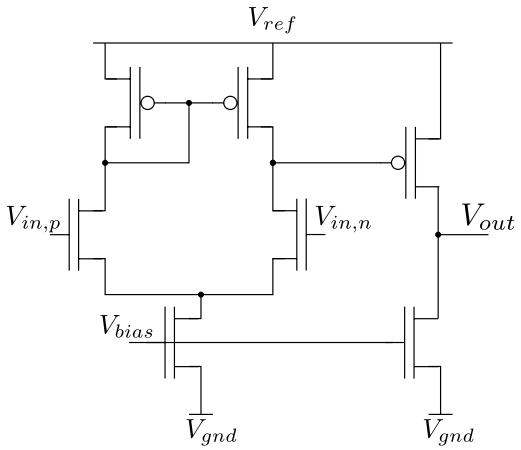


Figure 3.43: Two-stage open loop comparator [37]

– pre-amplifier based latched comparator – combination of open-loop comparator and a latch (example presented in Figure 3.44). Such combination allows for low offset (reduction of latched stage offset thanks to pre-amplifier's high gain) and reduction of both kickback and metastability problems (both phenomenons will be explained later). Commonly a clock signal is employed to change between operation modes – reset and evaluation. They also faster than open-loop comparators but static power consumption is still a problem.

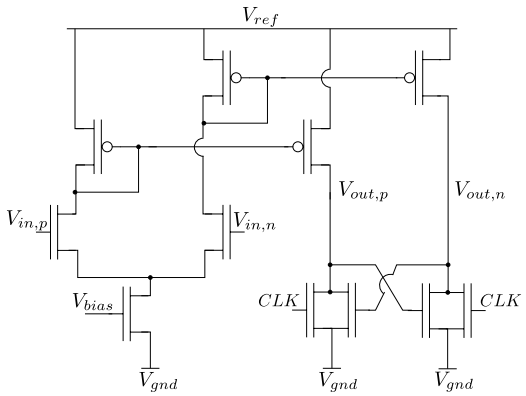


Figure 3.44: Static latched comparator [37]

– fully dynamic latched comparators – circuits from this group can work differently from one another (e.g. Lewis-Grey comparator [38] uses input transistors in triode mode as voltage controlled resistors while double-tail dynamic latched comparator [39] uses differential input pair and is separated into two stages), but they all have in common a latch output which in principle uses input-voltage dependant capacitance discharge time to resolve input level. Comparators from this group have high speed, full-swing output with high power-efficiency (not static power consumption). Clock signals are always used in this comparators to change from evaluation to reset phase and back. Since a comparator from this group is used in presented design a more detailed description of it is presented in section 4.3.

Kickback noise

An interesting phenomenon happening during operation of latched comparator is so called kickback noise. It's nature can be understood by considering a simplified structure of said comparator shown in Figure 3.45. Circuit proceeding comparator (in case of presented ADC it would be capacitive DAC) is modelled as voltage source with small series resistance. This source drives gates of input pair's transistors, which have some parasitic capacitances (e.g. between drain and gate, or source and gate). When a very fast switching from V_{gnd} to V_{ref} or from V_{ref} to V_{gnd} occurs (during evaluation phase) those parasitic's reactance $\frac{1}{2\pi fC}$ changes – the sharper the voltage change, the higher frequency components are presents, thus parasitic's reactance gets lowered. This can be approximated as a short between drain and gate of input transistor resulting in distortion of gate's voltage – so called kickback noise.

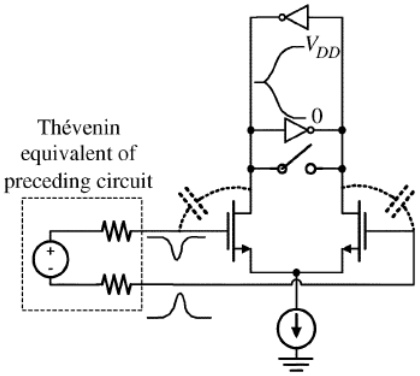


Figure 3.45: Kickback noise generation in latched comparator. [40]

In general faster comparators generate more kickback noise [40]. There are techniques to mitigate this effects, one of the more common is to add pre-amplifier before comparator, but a drawback of this solution is introducing static power consumption to circuit.

Metastability

Latched comparators, as was mentioned before, relay on input voltage dependant capacitance discharge time to resolve input level. This leads to longer comparator decision times for small input voltage differences. When this voltage difference is small enough comparator might have not enugh time to resolve given samples properly which might lead to bad interpretation by succeeding gates leading to a conversion error. In [30] it was shown that probability of such event occurring P_M can be described as:

$$P_M = \frac{1}{A_k}\frac{V_{ref}}{V_M}\mathrm{e}^{-\frac{T_{max}}{\tau}} \tag{3.32}$$

where A_k is comparator's gain factor, V_M is voltage range over which input signal changes, T_{max} is maximal time allowable for a comparator decision and τ is regeneration time constant of comparator.

In [41] a far more extensive examination of this phenomenon can be found, including derivation of minimal comparator's gain to allow for metastability errors impact be lower than that of quantization noise.

3.5 SAR logic

Because architecture of SAR logic depends mainly on incorporated algorithm and on required ADC parameters (e.g. conversion frequency), it will be described in detail in next chapter. Here only brief discussion of one of logic's main building blocks – D flip-flop – will be presented.

RST	CLK	D	Q_{next}
1	rising	0	0
		1	1
	non-rising	X	Q
0	X	X	0

Table 3.14: Example of a truth table for D flip-flop.

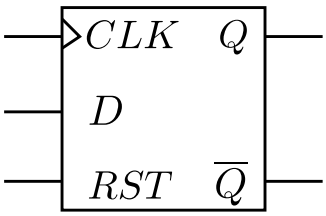
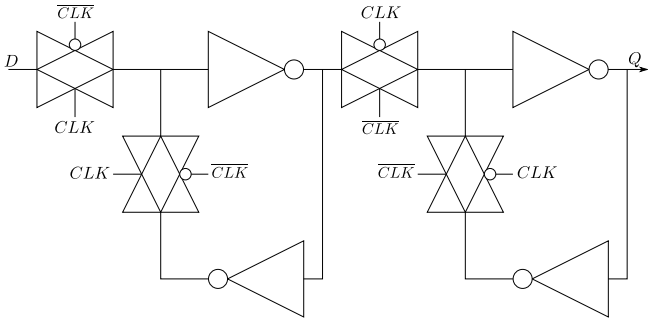
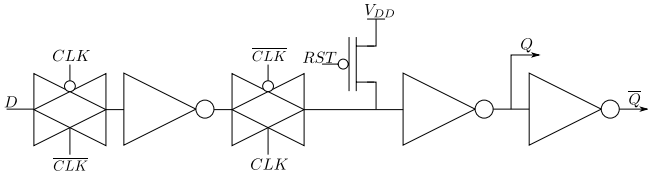


Figure 3.46: Symbol of D flip-flop.

D flip-flops (symbol and example of truth table presented in Figure 3.46 and Table 3.14) are used in all SAR algorithms implementation to construct chains of gates able to follow successive approximation algorithm. D flip-flop (DFF) can be constructed in many different ways depending on main goal of design – classical DFF (shown in Figure 3.47a) consisting of two latches connected in master-slave configuration is low design risk (with non-overlapping clock signal risk of race condition is minimal, transmission gate in front of feedback inverters prevents any fight between feedback and new input). Additional advantage is very low leakage of charge during absence of clock thanks to feedback inverters. If on the other hand speed is of main concern DFF can be constructed as show on Figure 3.47b – absence of feedback inverters and their transmission gate allows for faster operation and lower transistor count, though due to charge leakage output state will not be held for long time after clock signal disappears. More architectures are available e.g. in [43].



(a) Static D flip-flop.



(b) Dynamic D flip-flop with reset.

Figure 3.47: Used architectures of D flip-flops.

3.6 DAC switches

To achieve desired ADC resolution DAC switches must be sized in such a way to change bottom plate voltages quickly enough to allow for DAC's output voltage to settle within required accuracy within time that is allowed for 1 bit conversion. Assuming simple RC model of switch as resistance R_{sw} and corresponding to it DAC's capacitance C_{sw}, output voltage V_{out} will behave according to:

$$V_{out}(t) = V_{in}\left(1 - \mathrm{e}^{\frac{-t}{R_{sw}C_{sw}}}\right) \tag{3.33}$$

To achieve $V_{out} = V_{in} - 1\mathrm{LSB}$ in worst case scenario (starting with $V_{out} = V_{gnd}$ and having to charge DAC to $V_{in} = V_{ref}$) results in charging time Δt expressed as:

$$\Delta t = R_{sw}C_{sw}\ln\left(2^N - 1\right) \overset{N=12}{\approx} 8.3 \cdot R_{sw}C_{sw} \tag{3.34}$$

This useful equation allows for relatively easy sizing of DAC switches based on RC time constant of output voltage (C_{sw} is known and constant for every switch, therefore width of switching transistor must be increased to the size achieving desired RC constant of output voltage).

4 | Design of 12-bit SAR ADC

This chapter presents a precise description of SAR ADC designed in 130nm IBM CMRF8SF CMOS technology. Figure 4.1 shows a block diagram of converter (since presented work is focused on R&D few ADCs were designed, differing in used DAC architecture and resulting from that small differences in other building blocks; despite that block diagram for all ADCs is the same). Signal names shown in this figure will be kept throughout this chapter.

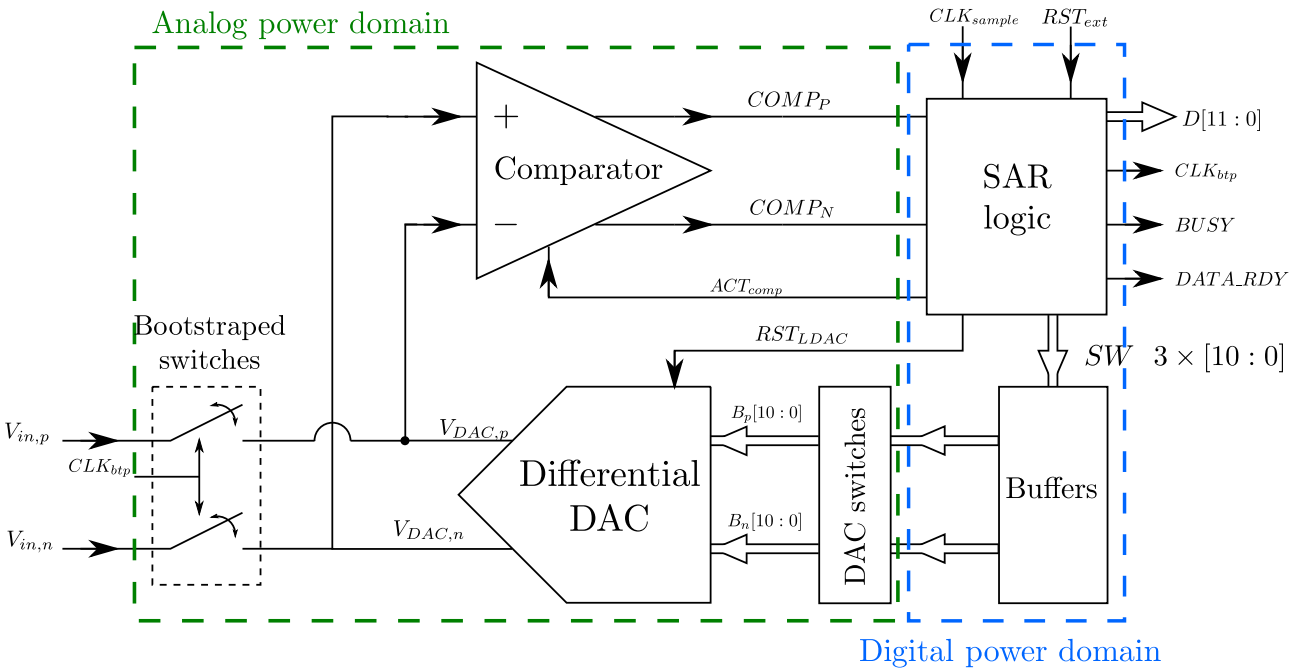


Figure 4.1: Block schematic of designed 12-bit SAR ADC.

Design of this ADC started with a VerilogA description of all building blocks (except for DAC which was in this starting phase built out of ideal capacitances), which were then gradually replaced by their transistor-level equivalents. Such approach allowed for easier design process than starting with transistor-level design for every block separately and then putting them together (easier functional verification, shorter simulation times – VerilogA code is simulated quicker than transistor-level schematics). The order in which those building blocks are described in this chapter is the same as the order in which their schematics were designed and is a result of dependence of one blocks on other circuits characteristics (e.g. bootstrapped switches transistors size depend on their load capacitance, which is total capacitance of DAC). Since

there are two power domains present in the design different names for sources for each domain will be used: $V_{ref,a}, V_{cm}, V_{gnd,a}$ for analog domain and $V_{ref,d}, V_{gnd,d}$ for digital domain.

4.1 DAC

Since the goal of this work is to design a 12-bit SAR ADC with as small power consumption and area as possible, split DACs with ratio M=8, L=3 and M=7, L=4 based on Table 3.12 appeared to be a good balance between area saving and relatively low mismatch influence. Next step was finding capacitor with good matching quality available in used technology.

4.1.1 MIM capacitor DACs

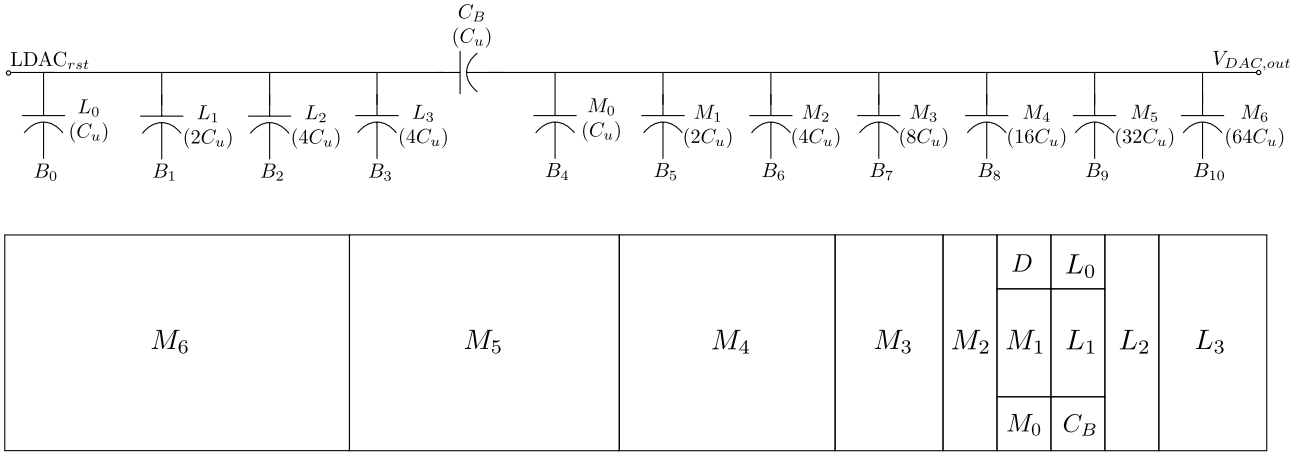
Based on design manual for used technology [31] MIM (metal-insulator-metal) capacitors appeared to have the best matching qualities among all available capacitors. Because design specification constrained pitch of designed ADC to $144\mu m$ minimal MIM-capacitors allowed by DRC rules were chosen as DACs building blocks – this allows for construction of two DAC placed side-by-side, each built out of four rows of capacitors. Such minimal MIM capacitor has matching coefficients: $K_C = 2.05\left[\frac{fF}{\mu m}\right]$, $K_\sigma = 4.12\,[\%\mu m]$, which means that, based on equation 3.23, unit capacitance should be:

- for split L = 4, M = 7: $C_u = 359.89fF$
- for split L = 3, M = 8: $C_u = 180.7fF$

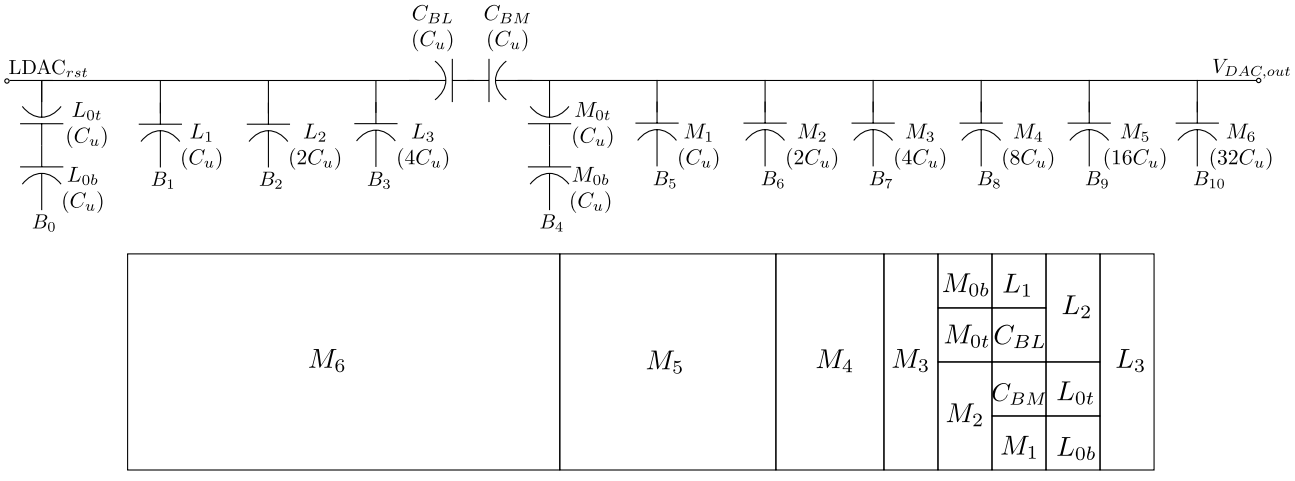
Those results indicate that actual capacitance of chosen MIM-capacitor ($60fF$) is far too small and ADC with DAC built out of such small capacitors will have very big performance variation due to mismatch (thermal noise is, based on equation 3.27, completely negligible). On the other hand capacitances as big as those calculated above are undesirable considering the limitations (e.g. $144\mu m$ pitch) and requirement to keep power consumption as small as possible. To check ADC's performance variation when using small unit capacitance three DACs were designed:

- split L = 4, M = 7: $C_u = 60$ (refereed to as L4M7 DAC, presented in Figure 4.2a)
- split L = 4, M = 7: $C_u = 30$ (refereed to as L4M7-0.5C DAC, presented in Figure 4.2b)
- split L = 3, M = 8: $C_u = 30$ (refereed to as L3M8-0.5C DAC, presented in Figure 4.2c)

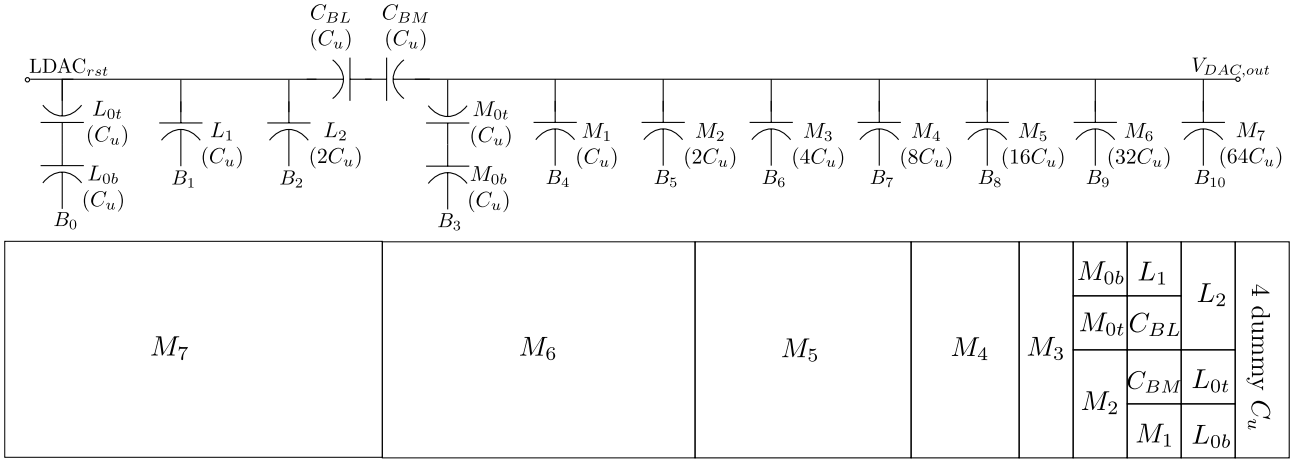
Unit capacitance of 30 fF is obtained by connection in series two 60 fF capacitors. To check ADC's performance variation 200 Monte Carlo simulations where done for three considered ADCs (using different DACs). To be sure that all effects are related to capacitors all ADC's blocks used for simulation, except for DAC, where VerilogA models, not transistor-level schematics. Simulation where done for 20MHz sampling and Discrete Fourier Transforms were calculated based on 64 samples (this relatively low number of samples is a result of long simulations times) using script made by staff of Department of Particle Interactions and Detection Techniques WFiIS AGH. Results are presented on Figures 4.3a, 4.3b and 4.3c.



(a) Schematic and layout floor-plan of L4M7 DAC.

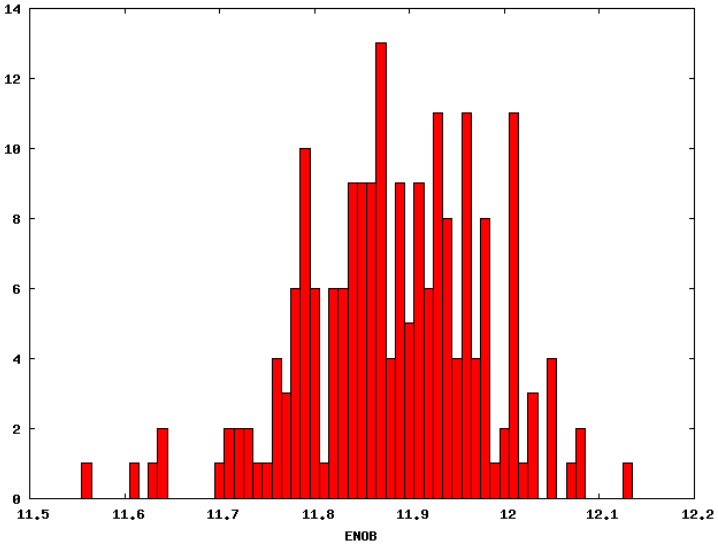


(b) Schematic and layout floor-plan of L4M7-0.5C DAC.

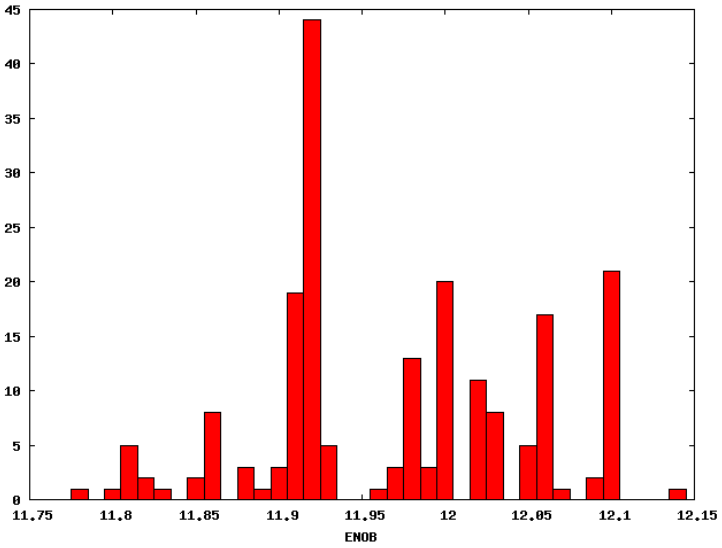


(c) Schematic and layout floor-plan of L3M8-0.5C DAC.

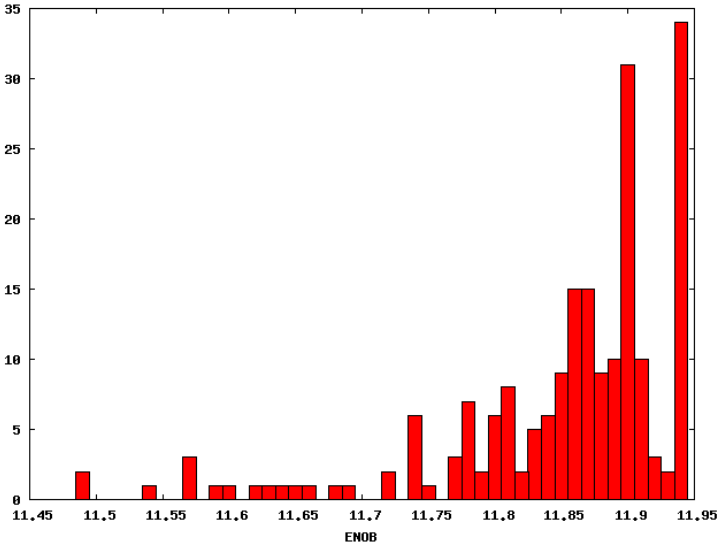
Figure 4.2: Schematics and layout floor-plans for MIM-capacitor based DACs.



(a) ADC with L4M7 DAC.

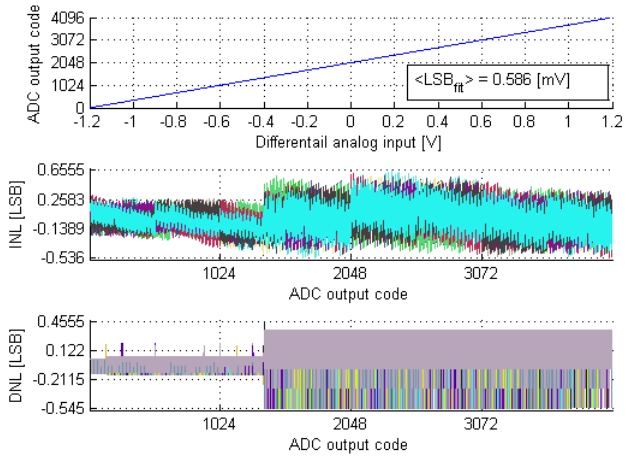


(b) ADC with L4M7-0.5C DAC.

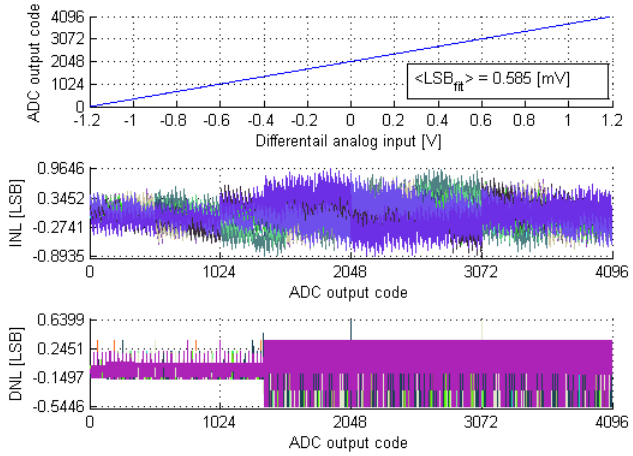


(c) ADC with L3M8-0.5C DAC.

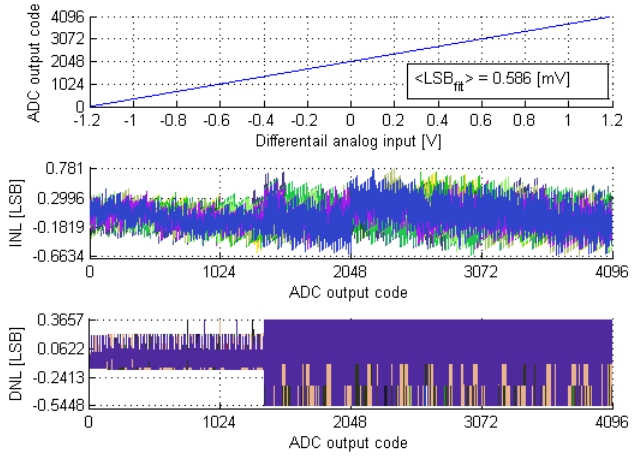
Figure 4.3: Results of 200 Monte Carlo dynamic simulations for ADCs with different DACs schematic (rest of the functional blocks – VerilogA).



(a) ADC with L4M7 DAC.



(b) ADC with L4M7-0.5C DAC.



(c) ADC with L3M8-0.5C DAC.

Figure 4.4: Results of 15 Monte Carlo static simulations for ADCs with different DACs schematic (rest of the functional blocks – VerilogA).

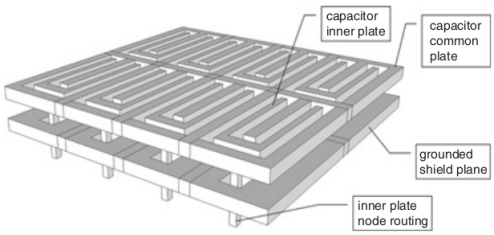
The performance variation was also checked with static simulations of ADCs with all three DACs (also with all functional blocks except for DAC replaced with their VerilogA models). Because of very long simulation times a standard approach like histogram method described in [16] could not have been adopted, so simulations were done by introducing a very slowly changing differential ramp signal to ADC's input and sampling it's output every 25ns (40MS/s). Rise time of ramp signal was chosen in such a way to (for ideal case) sample every LSB step three times (about 12000 data points per simulation were gathered). Simulation data was than analysed and input voltage of data points having the same ADC output were averaged. Obtained in this way an analog voltage value assumed was to be the middle of LSB step – after such analysis from about 12000 points initially gathered, a data set of 4096 points was obtained. From this a transfer curve of ADC was plotted and both INL and DNL are calculated in a way described in section 2.1.1. Since the number of samples per LSB step is relatively low, results presented on Figure 4.4a, 4.4b and 4.4c should be treated only as approximation of real performance.

Results presented on Figure 4.3 in all cases show that ADC's ENOB stays within 0.5-bit range of 12-bit value, which is an acceptable result (ENOB above 12 bits is most likely a result of relatively low accuracy of simulation – only 64 samples per DFT). Larger spread of ENOB for configurations using $C_u = 30fF$ is to be expected, but even in their case overall results are reasonably good. Also static simulations presented in Figure 4.4 are relatively good, there are no non-linearities which would indicate missing codes (DNL linearity errors larger than 1LSB), but all converters are not monotonic (INL above 0.5 LSB). It can be seen though that DAC with smaller unit capacitance perform worse. Higher errors observed in L4M7-0.5C DAC compared to L3M8-0.5C DAC despite both DACs having the same unit capacitance is also to be expected because of higher total number of capacitors in L3M8-0.5C DAC leading to lower influence of mismatch. This might suggest that equation 3.23 is too conservative and leads to unit capacitance values larger than in reality needed (assuming that the technology parametrization for MIM-capacitors is good).

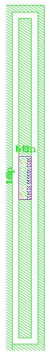
4.1.2 MOM-capacitor based DAC

One of possible alternatives to MIM capacitor is metal-oxide-metal (MOM) capacitor, which uses parasitic capacitances that exists between metal lines, as shown on Figure 4.5a. Although MIM capacitors use vertical electric field, which is denser than lateral field when per layer capacitance is considered, MOM capacitors can be stacked in several layers using both lateral and vertical field – in such configuration their density can be higher. MOM capacitors in form shown in Figure 4.5a are not available in used in this work design kit, but there are capacitors using the same principle – VNCAPs (example shown in Figure 4.5c) – built out of interleaved metal multi-finger structure. Problem with those capacitors is their big top and bottom plates (on Figure 4.5c seen as left and right plate) – their area is quite big compared to area of actual

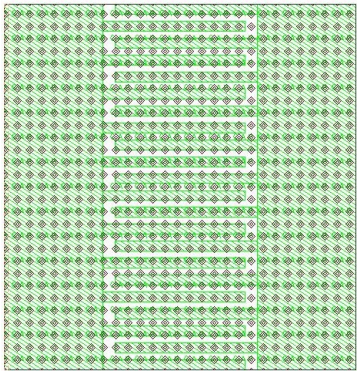
capacitor. This results in large undesired parasitic capacitance to substrate, which would make DAC designed out of VNCAP ineffective (unwanted parasitics would degrade output voltage levels). For those reasons designing MOM capacitors by drawing by hand structures like that presented in Figure 4.5b seemed beneficial – area of metal in relation to capacitor area is much smaller compared to VNCAP, which should lead to smaller value of unwanted parasitic capacitance. Additional benefit of making structures by hand is lack of restrictions on capacitor size (only restrictions for spacing of metal lines and their width remain). Such approach has also its drawbacks – due to construction of MOM capacitors it would be rather difficult to build split DAC out of them, so to minimize number of used capacitors a classical (not split) binary-weighted MOM DAC was build to suite AMCS algorithm requirements (this of course requires some modifications in SAR logic). Also no models for Monte Carlo simulations are available for capacitors designed in described way, which means that reliability of MOM-capacitor based DAC cannot be checked through simulations.



(a) MOM capacitor array [33]



(b) Example of single MOM capacitor.



(c) Example of single VNCAP capacitor.

Figure 4.5: Examples of MOM and VNCAP capacitors (not to scale).

To find the best configuration of metal layers for MOM-capacitors few different attempts were made, but all of the suffered the same flaw – very high unwanted parasitic capacitance to substrate, it's value ranged from 30% to 50% of unit capacitance. Best result – $C_u = 2.5fF$) with $C_{parasitic} = 0.8fF$ – was obtained for capacitor presented in Figure 4.5b constructed out of Metal3 (outer ring (common for all capacitors top plate) $1.2\mu m \times 10.8\mu m$, inner metal strip (bottom plate) $0.2\mu m \times 10\mu m$) with no additional shielding. Routing was done using Metal5 (MQ). Schematic and layout floor-plan of DAC based on this unit capacitor are presented in Figure 4.6. No common-centroid technique is applied to keep amount of metal used for routing to minimum (to minimize undesired parasitics capacitances between metals).

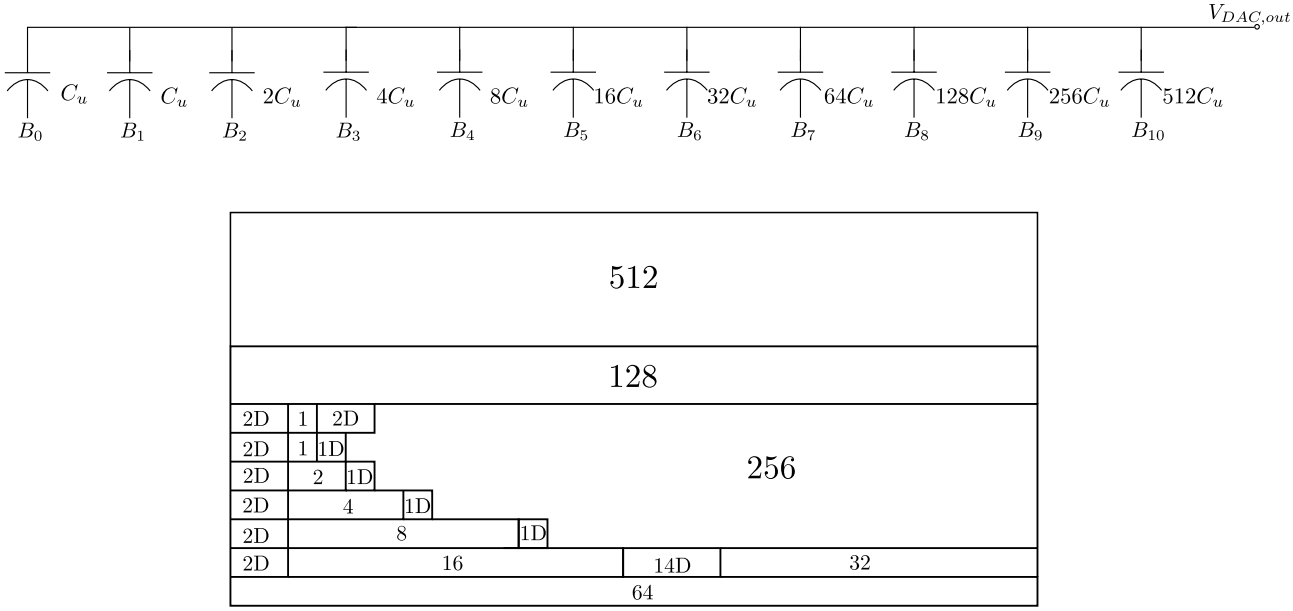
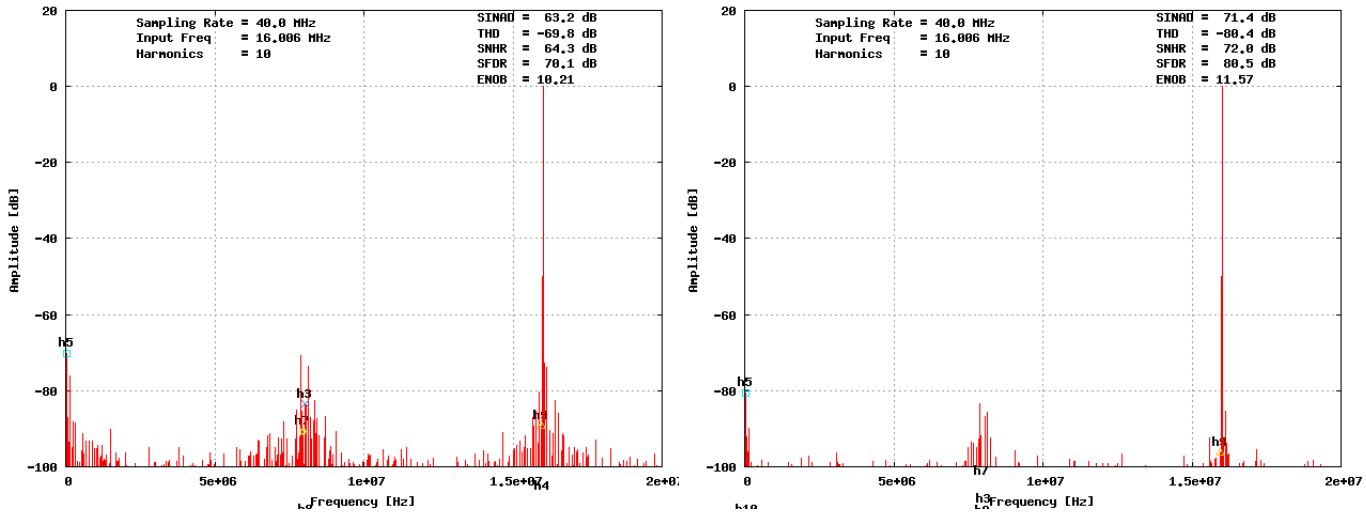


Figure 4.6: Schematics and layout floor-plan for MOM-capacitor based DAC (capacitor sizes are not to scale). Letter D indicates dummy capacitors.

Even in this best case though performance degradation is observed – additional parasitic capacitance to substrate results in capacitive division of each unit capacitor voltage. This means that e.g. effective reference voltage for DAC is lower than actual one and therefore input signals with large amplitude saturate converter. For described MOM-capacitor differential signal's amplitude had to be lowered from 1.15V (as used for MIM-capacitor based DACs) to 0.88V in order to not saturate ADC during simulations. Because no Monte Carlo model of inter-metal parasitic capacitances was available only single measurement of ADC performance was done. Similarly to MIM-capacitor based DACs all building blocks in ADC were substituted for VerilogA models. Simulations were done for both purely capacitive extract of DAC and fully RC extract (both parasitic resistances and capacitances included in simulation), DFT was calculated based on 4096 samples – results are presented on Figure 4.7. Reason for much better results when simulating circuit with full RC extracted has not been found.



(a) Results for DAC with only capacitive parasitics.

(b) Results for DAC with capacitive and resistive parasitics.

Figure 4.7: ADC performance with MOM-capacitance based DAC (differential input amplitude lowered to 0.88V).

Static simulations (performed in exactly the same way as for MIM-capacitor based DAC, only narrowing analysed analog voltage range to not saturate converter) are presented in Figure 4.8.

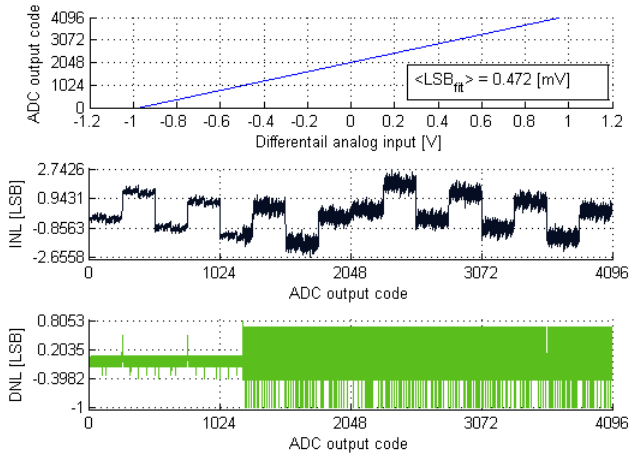


Figure 4.8: Results static simulation (obtained through simulation) for ADCs with MOM-capacitor based DAC extract (rest of the functional blocks – VerilogA).

Figures 4.7 and 4.8 show that with lowered maximal input signal amplitude ADC works quite well during dynamic simulations, but static performance is much worse (very non-linear behaviour with INL exceeding 1LSB for most of codes). Additional benefit of using MOM-capacitor based DAC is it's much smaller size compared to MIM-capacitor based converters, as is presented in Table 4.1 and Figure 4.9.

Table 4.1: Size comparison of layouts of designed DACs.

DAC	Width [μm]	Length [μm]
MOM-based	146	188
MIM-based L4M7-0.5C	144	435
MIM-based L4M7		815
MIM-based L3M8-0.5C		800

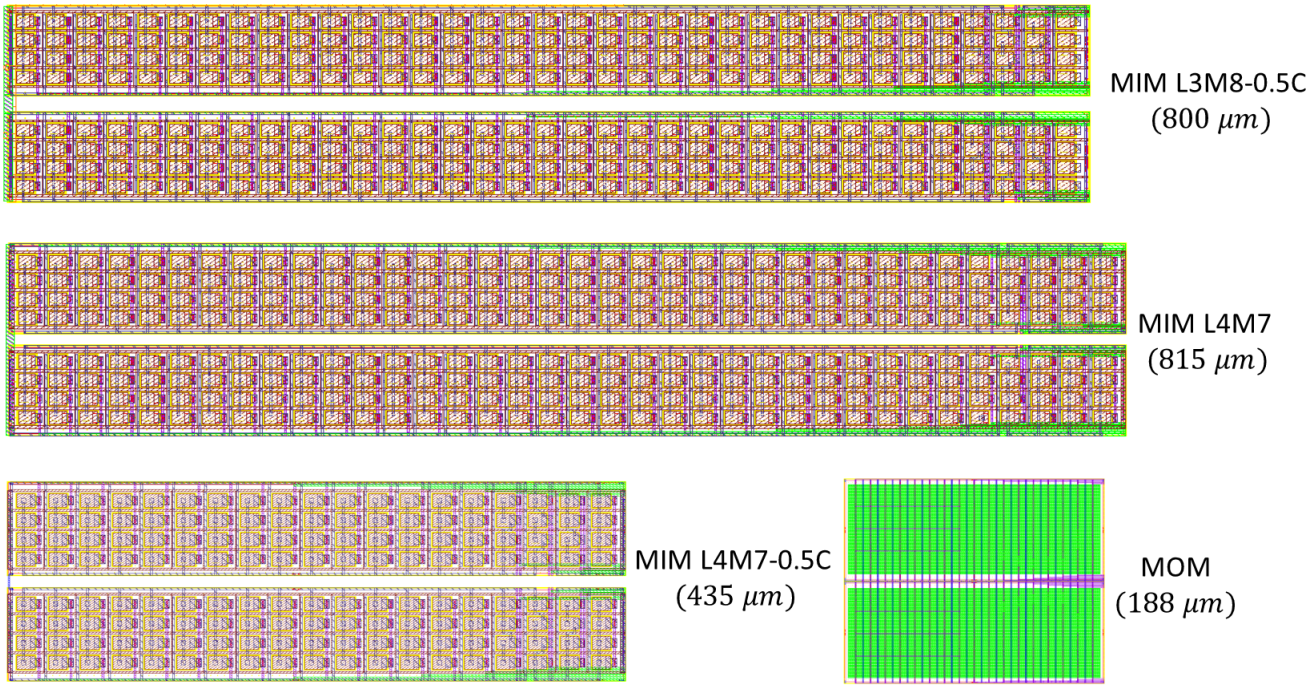
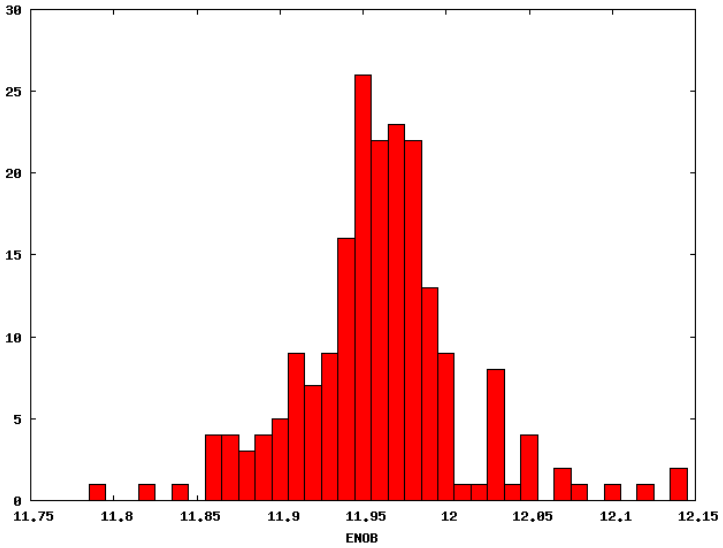


Figure 4.9: comparison of layouts of designed DACs.

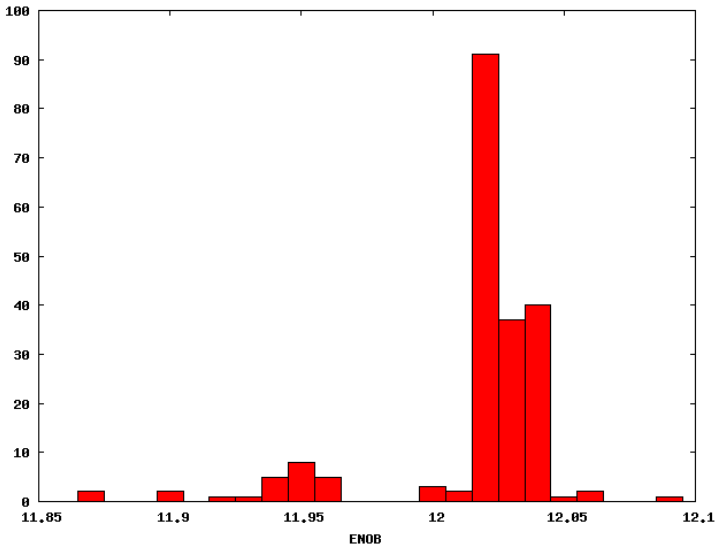
4.1.3 Comparison of MCS and EMCS algorithms influence on DAC's performance

While MCS algorithm was chosen instead of EMCS to be implemented in presented design (for various reasons), EMCS has one very interesting feature – thanks to eliminating the need for worst case code switching (e.g. $[01\cdots11] \rightarrow [10\cdots00]$) linearity of ADC should improve (when comparing to same ADC's configuration using MCS algorithm). To see how large said improvement would be in case of 12-bit ADC, simulations using VerilogA model of functional blocks (including EMCS logic) and schematics of two of designed DACs were carried out (both dynamic and static, simulation methodology was also exactly the same as in previous sections). Results are presented in Figure 4.10 and 4.11. Comparing Figures 4.3a with 4.10a, and 4.3c with 4.10b reveals that variation of values of ENOB is smaller when using EMCS algorithm, while

analysis of static simulations (comparison of Figure 4.4a with 4.11a, and 4.11b with 4.11b) shows about 20% lower INL (DNL is unchanged). This results are very encouraging (quite substantial improvement in performance by just changing the algorithm of DAC switching) and might lead to re-consideration of implementing EMCS algorithm as a potential future work on the project.

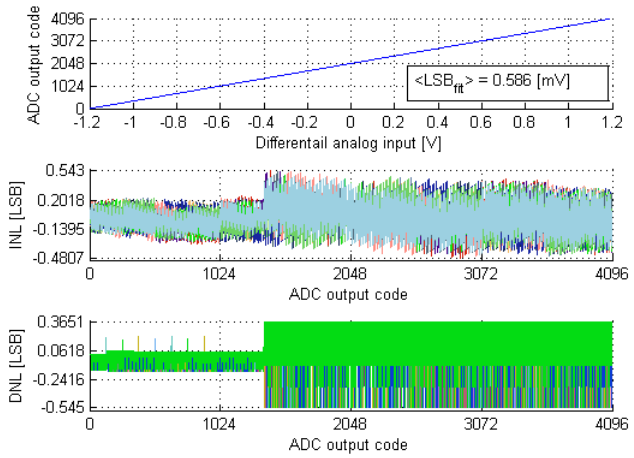


(a) ADC with L4M7 DAC.

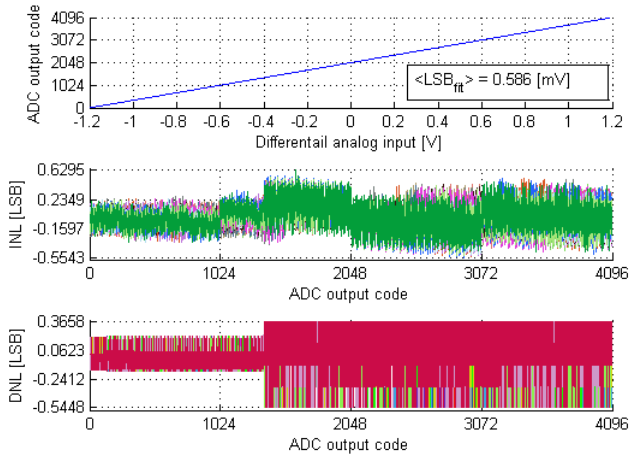


(b) ADC with L3M8-0.5C DAC.

Figure 4.10: Results of 200 Monte Carlo dynamic simulations for ADCs (using EMCS algorithm) with different DACs schematic (rest of the functional blocks – VerilogA).



(a) ADC with L4M7 DAC.



(b) ADC with L3M8-0.5C DAC.

Figure 4.11: Results of 15 Monte Carlo static simulations for ADCs (using EMCS algorithm) with different DACs schematic (rest of the functional blocks – VerilogA).

4.2 Bootstrapped switch

Transistor-level schematic of bootstrapped switch (based on [36]) is presented in Figure 4.12 (transitor's bulk connections are marked only in non-standard cases i.e. nMOS bulk at potential different than $V_{gnd,a}$ or pMOS bulk potential different than $V_{ref,a}$). Transistors $TN_1, TP_2, TN_3, TP_4, TN_5$ correspond to switches S_1, S_2, S_3, S_4, S_5 from Figure 3.39b, additional transistors are needed for more reliable operation:

- gate of TP_4 must be connected to node G (gate of $TNSW$) to be always able to turn transistor off when CLK is high (important for cases when input voltage is near to V_{ref} – potential at nodes B and G rises to $2V_{ref}$, so transistor could not be turned off if its gate would be connected to CLK)
- transistor TN_6 is used to connect gate of TP_2 to node A when CLK is low (if gate of

TP_2 was controlled by CLK than for input near $V_{ref,a}$ and low CLK signal gate-source voltage of TP_2 would be near $-2V_{ref,a}$ which might damage the transistor). When CLK goes high gate of TP_2 is connected to $V_{ref,a}$ by TP_7

- because gate of TN_6 is connected to node G, which potential is controlled by TP_2 a dependency loop exist between the two transistors. For this reason TN_{6S} is needed to force start TP_2 to conduct when CKL goes high
- transitor TN_{T5} was added to prevent V_{GD} of TN_5 reaching $2V_{ref,a}$ when CLK is low

	for 4pF	for 8pF
WNSW [μm]	15.36	25.8
TN_1[μm]	0.48	0.48
TP_2 [μm]	0.48	0.48
TN_3 [μm]	0.48	0.48
TP_4 [μm]	0.48	0.48
TN_5 [μm]	5.76	7.88
TN_{T5} [μm]	5.76	7.88
TN_6 [μm]	0.96	0.96
TN_{6S} [μm]	0.96	0.96
TP_7 [μm]	0.48	0.48
C_{offset} [fF]	500	750

Table 4.2: Bootstrapped switches components sizes.

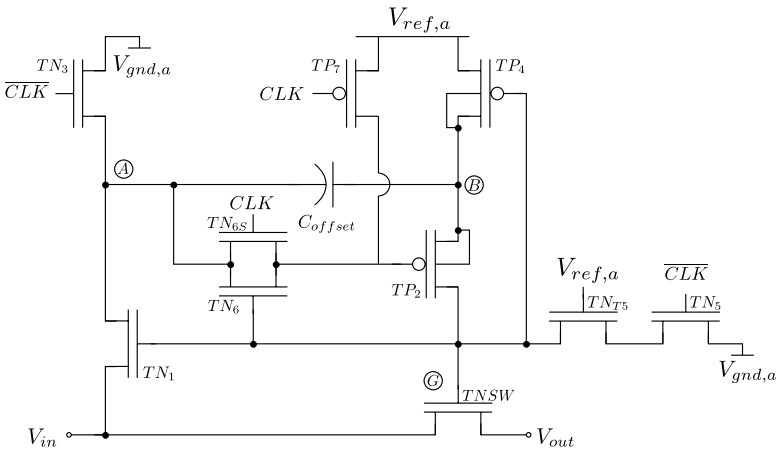


Figure 4.12: Transistor-level schematic of bootstrapped switch [36].

Sampling duration

Because four different DACs are designed in this work they require different bootstrapped switches. Sizing of transistor depends on bootstrapped switch's output capacitance (input capacitance of DACs), which are (for MIM-capacitor based DAC calculation is based upon Table 3.13, for MOM-capacitor based DAC simulated value is presented):

- for L4M7 DAC $C_{in} = 8.28pF$
- for L3M8-0.5C DAC $C_{in} = 7.68pF$
- for L4M7-0.5C DAC $C_{in} = 4.14pF$
- for MOM DAC $C_{in} = 3.31pF$

This shows that only two separate bootstrapped switches are needed because there are two pairs of DACs with similar input capacitance. All transistors in designed bootstrapped switches have length of 120nm (minimal allowable in used technology), widths are presented in Table 4.2.

Figure 4.13 shows simulated sampling accuracy (in bits) as a function of sampling time duration - it can be observed that at least 5ns is needed to sample input with enough accuracy

(13bits, to be sure that switches do not limit accuracy of the ADC), but in final design 7ns sampling time was chosen (for both switches) to get additional safety margin.

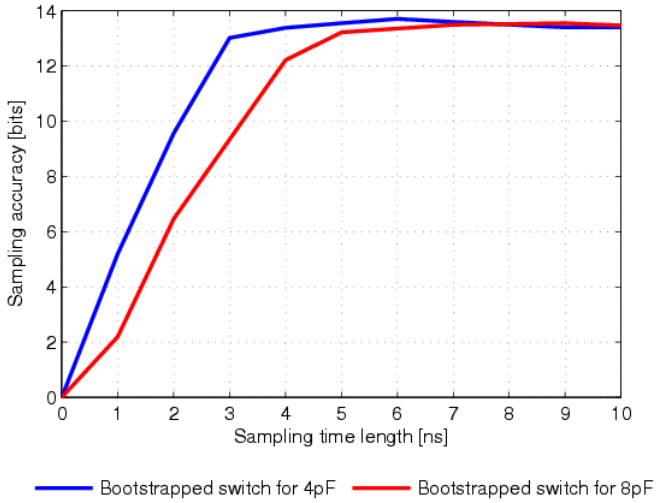


Figure 4.13: Sampling accuracy of designed bootstrapped switches as a function of sampling time.

4.3 Comparator

A fully dynamic latched comparator presented in [37] was used in the design because of its low offset and kickback noise, combined with fast operation. Schematic of circuit is presented in Figure 4.14 and all transistors widths are given in Table 4.3 (all transistors have length of 120nm).

TN_1	8
TN_2	12
TN_3	12
TP_4	6
TP_5	6
TN_6	4
TN_7	4
TP_8	4
TP_9	4
TP_{10}	3
TP_{11}	3
TN_{12}	6
TN_{13}	6
TP_{14}	1
TP_{15}	1
TN_{16}	3
TN_{17}	3
TP_{18}	9
TP_{19}	9

Table 4.3: Widths (in μm) of all the transistors used in comparator.

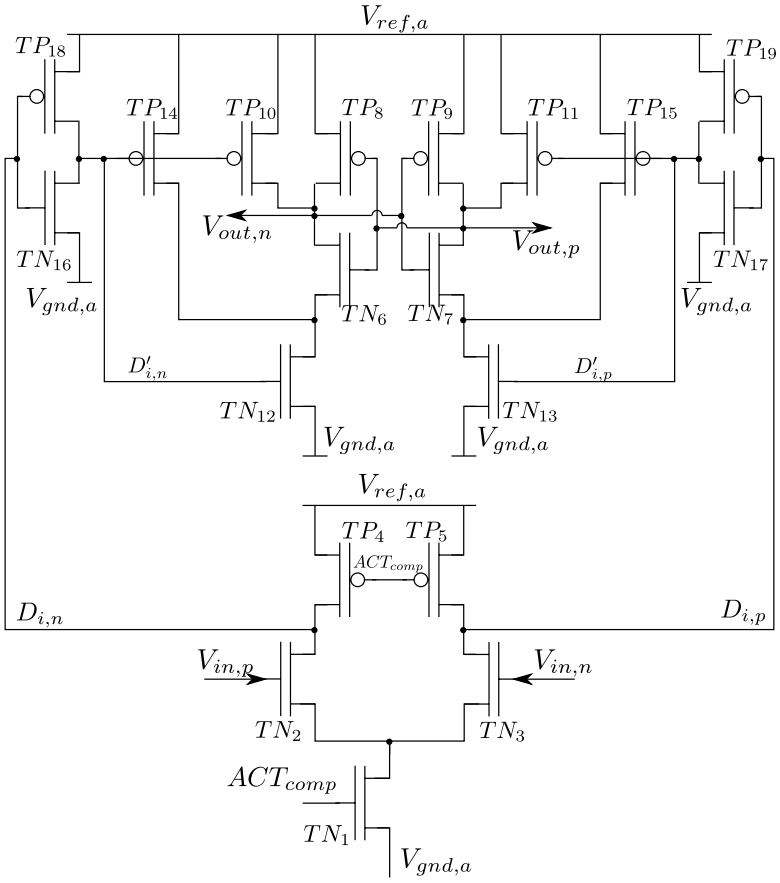


Figure 4.14: Transistor-level schematic of used comparator [37].

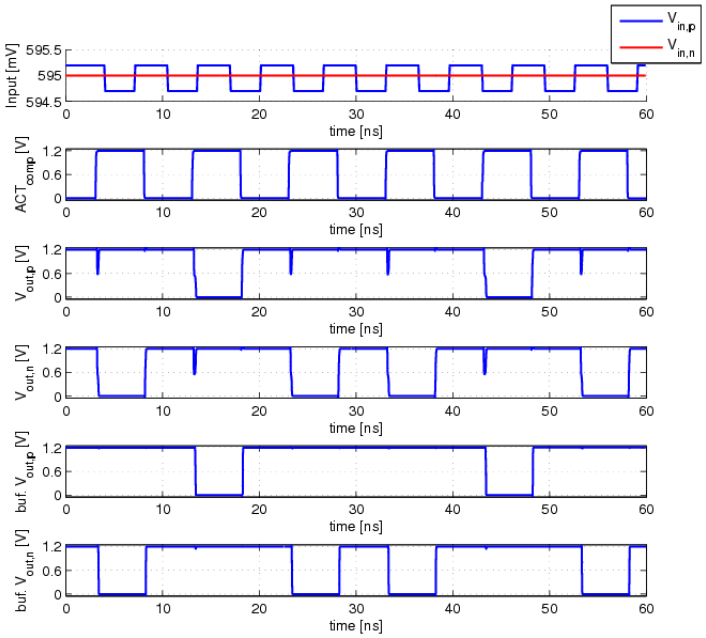
Simulated waveforms of presented comparator are shown in Figure 4.15. Operations of this circuit can be divided into two phases: reset and evaluation.

During reset phase ($CLK = 0$) transistors TP_4 and TP_5 are turned on so nodes $D_{i,n}$ and $D_{i,p}$ are charged to $V_{ref,a}$. This result in turning on transistors TN_{16} and TN_{17} leading to discharge of nodes $D'_{i,n}$ and $D'_{i,p}$ to $V_{gnd,a}$. This results in turning on transistors TP_{10}, TP_{11}, TP_{14} and TP_{15} and as result both $V_{out,p}$ and $V_{out,n}$ are reset to V_{ref}.

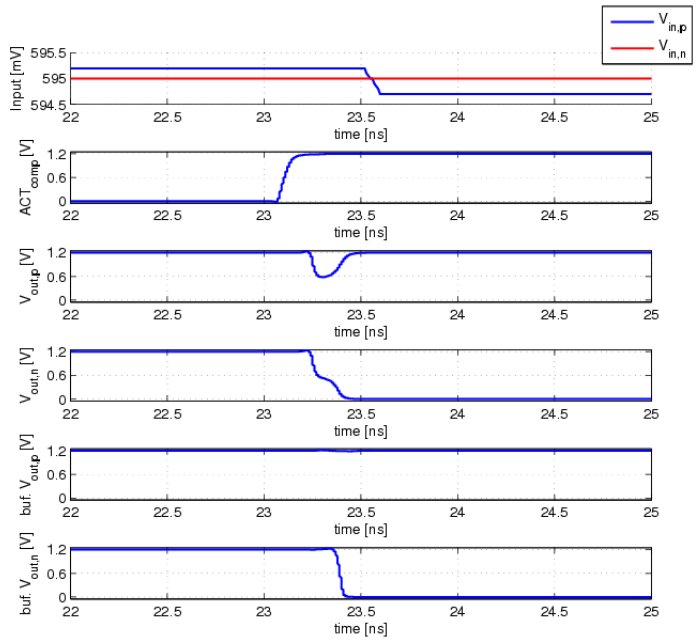
When CLK changes to logical 1 evaluation phase starts. Transistors TP_4 and TP_5 are turned of and $D_{i,n}$ and $D_{i,p}$ nodes are discharged to $V_{gnd,a}$ in rates dependant upon input voltage level. When either of the two voltages drops below $V_{ref,a} - |V_{Th,p}|$ (where $V_{Th,p}$ is threshold level of pMOS transistor) transistor TP_{18} or TP_{19} invert appropriate D_i node's voltage into D'_i node. As $D'_{i,n}$ and $D'_{i,p}$ rise within different times, they turn on one after the other TN_{12} and TN_{13} leading to start of latch regeneration at both outputs at different times. After either one of $V_{out,p}$ or $V_{out,n}$ drops below $V_{ref,a} - |V_{Th,p}|$ the positive feedback becomes much stronger (TP_8 and TP_9 are switched on). Output $V_{out,p}$ will be logical 1 if $D'_{i,n} > D'_{i,p}$ or will drop to logical 0 if opposite is true.

As can be seen from Figure 4.15a designed comparator resolves correctly voltages differing by 0.25mV ($\frac{1}{2}$ LSB). Due to metastability phenomenon described in previous chapter, time which is needed for this comparison is longer than in case of higher input voltage difference.

As can be seen on waveforms in Figure 4.15b around 200 ps is needed to perform comparison in case of so small input voltage difference to get correct results – this should be fast enough for designed ADC.



(a) Several comparisons.



(b) Close-up of one of the comparisons from (a).

Figure 4.15: Simulated waveforms of comparator operations.

4.4 DAC switches

Each of the binary scaled capacitors in both DACs (in each ADC) has it's own switch and associated with it buffers – Figure 4.16 presents one of such circuits.

Size of transistors in switch must be chosen large enough to ensure DAC's voltage settling with acceptable accuracy within time constrains based on conversion rate, which for this design is 40MS/s (meaning that there are 25ns between subsequent conversions). Taking into account that it was decided to allocate 7ns for sampling, only 18ns remain to determine 12bits, which means that on average one bit should be resolved every 1.5ns. Within this time digital logic must interpret comparator's output and decide next switching step, DACs must be switched and their voltage settle with high accuracy, so comparator can make correct decision. Simulations have shown that to achieve all that even in bad conditions (e.g. comparator's metastability) in given time DAC switching should not take longer than 0.6ns. Based on that information and equation 3.34 each switching transistor should be sized to such width (length was always kept minimal) that measured time constant $R_{sw}C_{sw}$ of DAC's output voltage is about 72ps (to allow for some safety margin actual sizing was done for 65ps).

After switches sizing was complete buffers that were needed to effectively drive switches were calculated (both number of buffers and their size) based on logical effort method described in appendix B. Example of simulated operation of switch with buffers obtained in described way is presented in Figure 4.17.

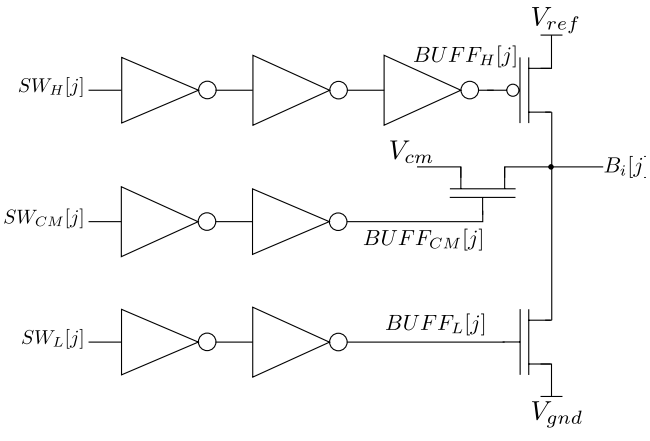


Figure 4.16: Switch with buffers for one of the capacitors of DAC.

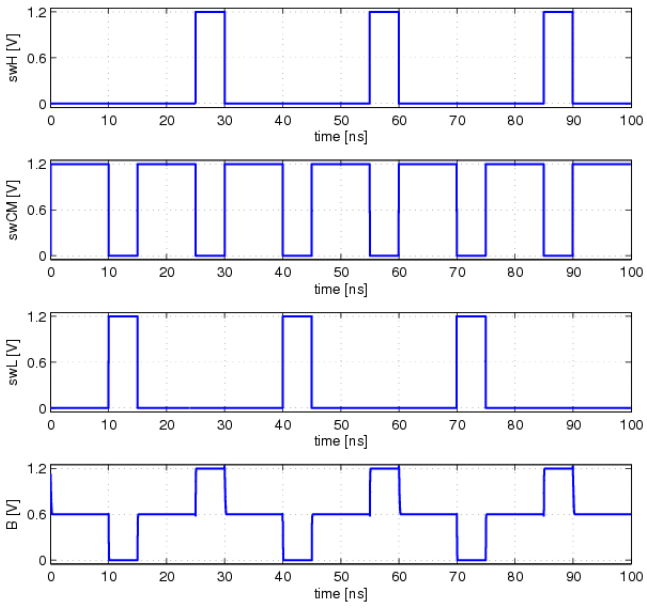


Figure 4.17: Simulated waveform of operations of DAC switch.

4.5 SAR MCS Logic

Designed logic is asynchronous – it does not need any external clock to control timing (apart from CLK_{sample} which just starts the conversion and does not control ADC after that in any way). As was explained in previous section average time for resolving one bit must be 1.5ns or less. This is equivalent to working with frequency of approximately 670 MHz – this means that logic cannot be synthesized automatically from Verilog code (using special tools) because circuits obtained in this way are build out of library elements, which would be too slow. As a result whole logic had to be done by hand, including design of logic gates and D flip-flops.

Successive approximation logic used in presented design consists of four main functional blocks as shown in Figure 4.18. Description of each block will be presented in separate section but signal names are kept the same throughout this chapter. Additionally all buffers used just to speed up signals will be omitted in presented schematics to keep them simple.

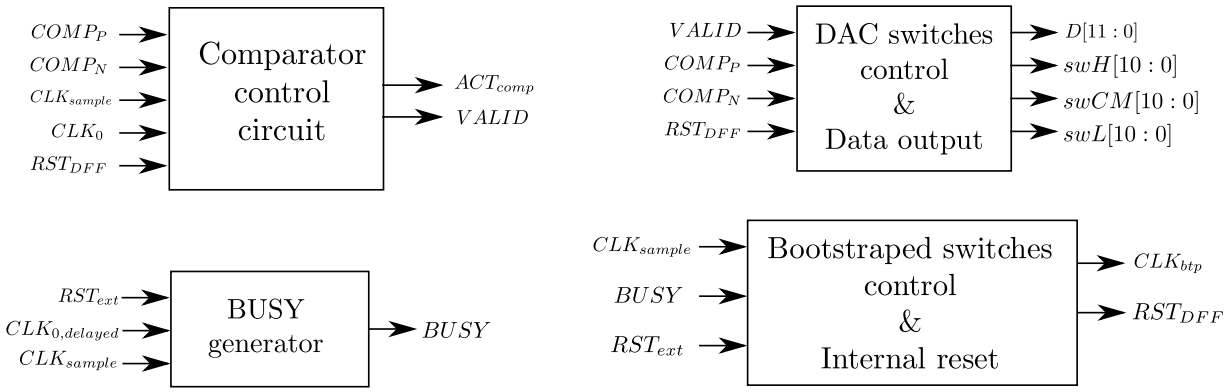
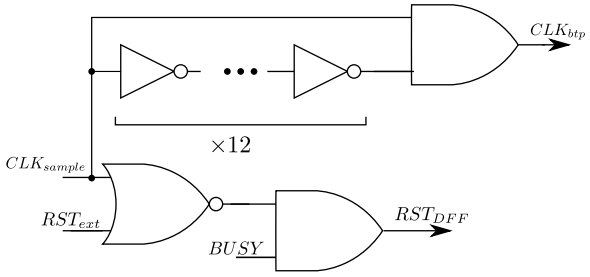


Figure 4.18: Block schematic of designed SAR MCS logic.

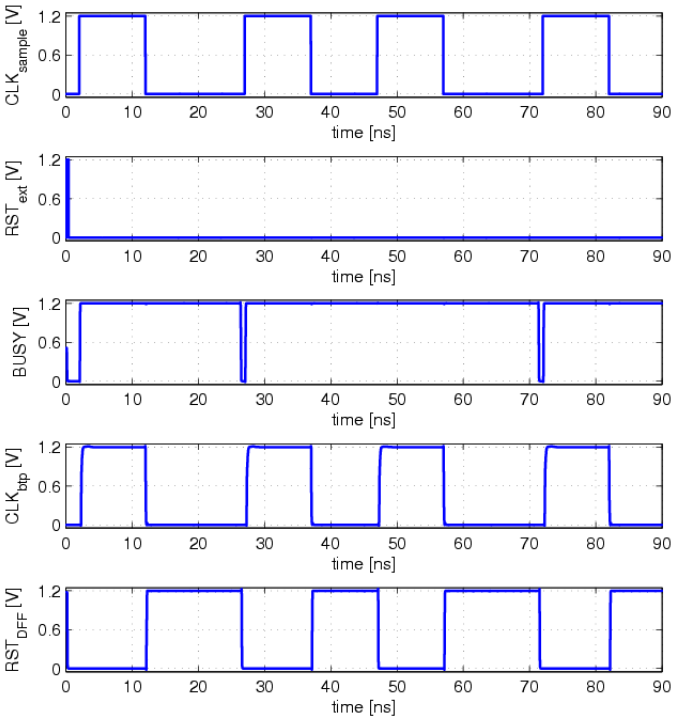
Bootstrapped switches control & internal reset

Simplified schematic of this block is presented in Figure 4.19a and simulated waveforms of its behaviour are shown in Figure 4.19b. This block of digital logic has two goals:

- produce CLK_{btp} signal which is used to control opening and closing of sampling switches. This signal goes high (starting sampling) 280ps after CLK_{sample} rises to have some time to reset the rest of the logic. When CLK_{sample} falls to zero, CLK_{btp} follows immediately so that sampling end is in fact controlled by CLK_{sample}.
- generate internal reset signal RST_{DFF} – reset is active (gates and DFFs are reset) when RST_{DFF} is low, so whenever one of signals CLK_{sample}, RST_{ext} (external reset) or $BUSY$ (indication that ADC is currently converting) goes high , reset for digital logic is stopped. In case of next sampling clock rising before current conversion is finished ($BUSY$ would still be high, as can be seen in Figure 4.19b at 48ns) ADC is reset and new conversion starts (any result from unfinished conversion is discarded).



(a) Simplified circuit schematic



(b) Simulated waveforms (with example of bad conversion from 30ns to 47ns)

Figure 4.19: Bootstrapped switch control and internal reset generator circuit with simulated waveforms of its operation.

DAC switches control & data output

Simplified schematic of this block is presented in Figure 4.21. This is the biggest and most complex block of digital logic. It consists of three chains of D flip-flops:

1. Control chain – task of this part of logic is to assure that:
   - DACs switching is performed in correct way (defined by MCS algorithm)
   - output bits D are resolved from MSB to LSB in correct order.

   Output of each DFF in this chain is connected to data input of next DFF in this chain (with the exception of first D flip-flop, which input is connected to V_{ref}) and to clock input of one of the decision circuit of decision chain. All DFFs in control chain have common clock input which is connected $VALID$ signal (it goes high after comparator decision is made and goes low when comparator is reset) – this means that after first comparator decision all DFFs in this chain are clocked but only first one's output – CLK_{11} – goes from 0 to 1 (at start all flip-flops are reset, so all starting outputs are at 0). When next comparator decision will be observed again all DFFs will be clocked and than output of first DFF will remain at 1 and output of second DFF – CLK_{10} – will change to 1, while the rest will remain at 0. Simulation results of this process are presented in Figure 4.20 (only few CLK signals are shown for clarity of plot).

   Each CLK is a triggering signal for one of the decision circuits from decision chain, so

the order in which CLK signals go from 0 to 1 decides the order in which decision circuits are used. This dependency assures that output bits will be resolved in proper sequence (from MSB to LSB).

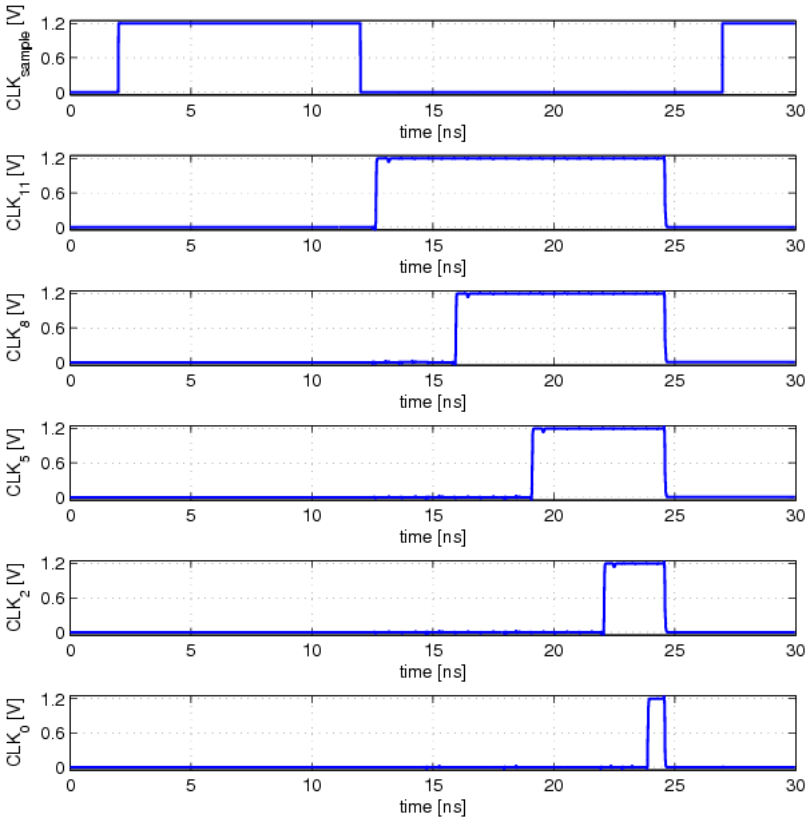


Figure 4.20: DAC switches control for MCS algorithm – circuit schematic.

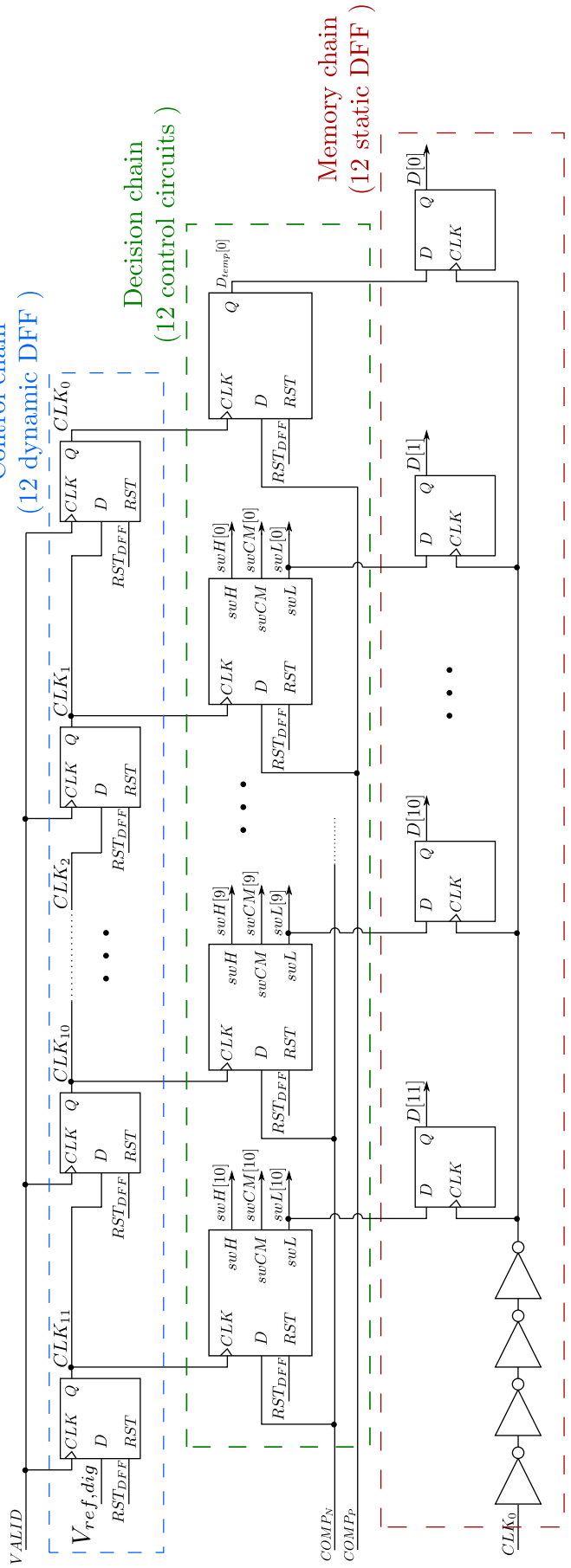
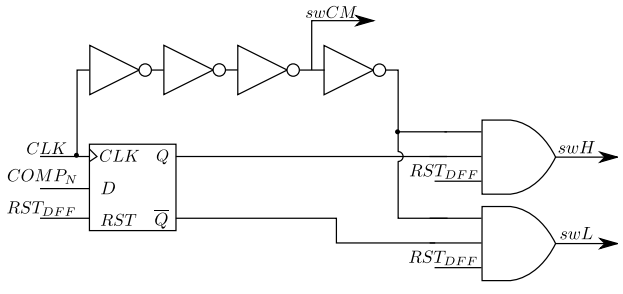


Figure 4.21: DAC switches control for MCS algorithm – circuit schematic.

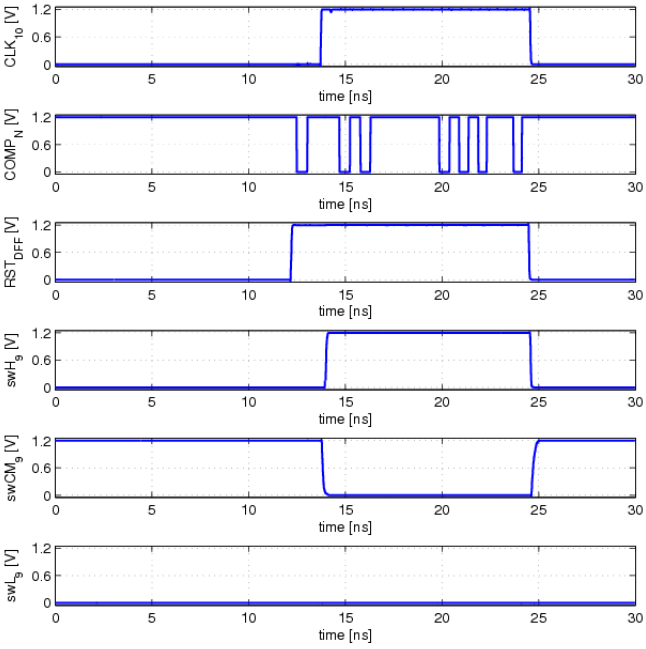
2. Decision chain – this part of digital logic consists of 12 decision circuits presented in Figure 4.22a and it's task is to determine to which potential capacitors from the DACs should be connected at given phase of conversion. When an i-th decision circuits is triggered by a CLK_i it changes states of $swH_i, swCM_i$ and swL_i – signals controlling to which potential 2^iC_u in DAC is connected (swH controls connection to $V_{ref,a}$, $swCM$ controls connection to V_{cm} and connection to $V_{gdn,a}$ is controlled by swL) If any of this signals is high, a potential with which this signal is bound is connected to DAC's capacitor:
   - $swCM_i$ is always an inversion of CLK_i, so when CLK_i goes to 1, $swCM_i$ goes to 0 resulting in disconnecting 2^iC_u in DAC from V_{cm}
   - if input of decision circuit is $COMP_P$ (p-side output of comparator) than Q output of DFF decides state of swL_i while swH_i is decided by $\overline{Q}$. If signal connected to DFF's input D is $COMP_N$ (n-side output of comparator) than Q controls swH_i and $\overline{Q}$ controls swL_i. Half of decision circuit are connected to $COMP_P$ and the other half to $COMP_N$ – reason for this is to attempt to even out capacitive loads of comparator's outputs since any mismatch in this respect might degrade comparator's performance [42].

   One of the inputs of AND gate controlling state of swL_i and swH_i signals is delayed CLK_i – this is implemented to assure that capacitors are disconnected from V_{cm} before being connected to $V_{ref,a}$ or $V_{gnd,a}$.

   Value of output bit D_i for given conversion is the same as state of swL_i, therefore swL signals are provided to memory chain as conversion results.



(a) Simplified circuit schematic.



(b) Example of simulated waveforms.

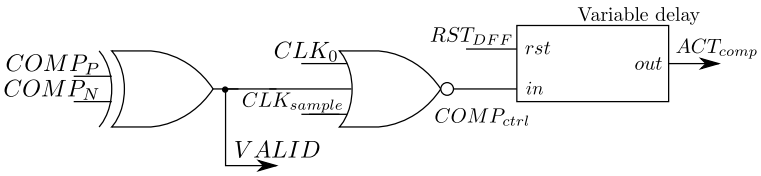
Figure 4.22: Decission circuit schematic together with simulated waveforms.

3. Memory chain – to speed up operations of both control and decision chains dynamic D flip-flops were used in both of them (schematic of such DFF is presented in Figure 3.47b). As a consequence any data stored in those DFFs might get corrupted due to charge leakage after being stored for long time. To remedy this memory chain was introduced consisting of 12 static DFF (with schematic as presented in Figure 3.47a). Input data for those flip-flops are ADC's output bits values found by decision chain. All DFFs in this chain are triggered by delayed CLK_0 signal, which means that ADC's output word is saved only after whole conversion is done.

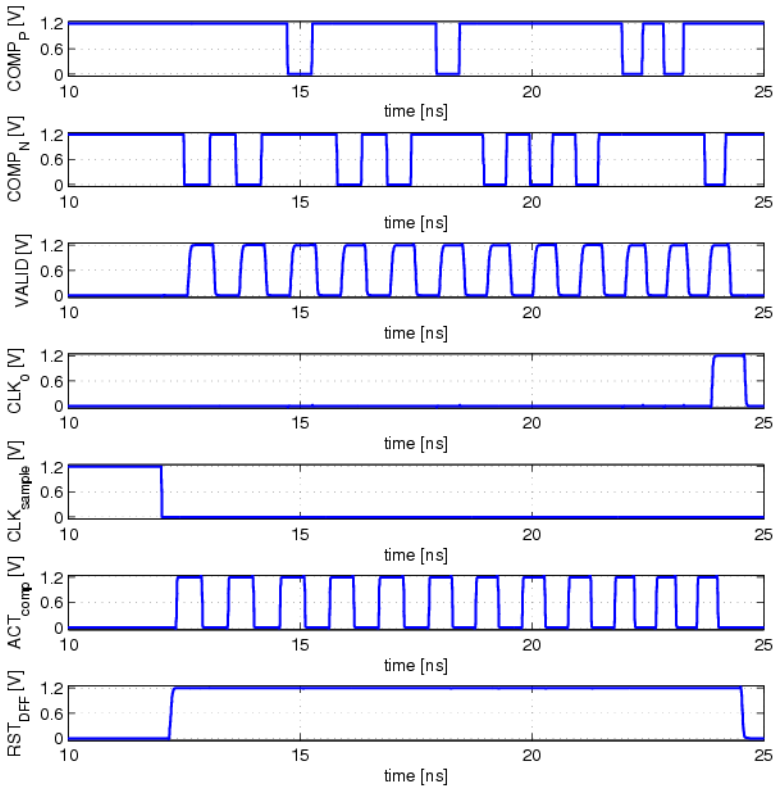
Comparator control circuit

Simplified schematic of this block is presented in Figure 4.23a, while Figure 4.23b presents waveforms from simulations of this block. This circuit has two tasks:

- generate $VALID$ signal which is used to monitor when a stable comparator output is available – $VALID$ goes to logic 1 only when one of comparator's outputs is high while the and other is low. If both of comparator's output are is the same logical state $VALID$ is at 0. This functionality is achieved by using single XOR gate.
- generate ACT_{comp} signal which controls comparator's reset and evaluation phases timing. This functionality is provided using 3-input NOR gate to generate $COMP_{ctrl}$ signal which is delayed by variable (between conversion phases) amount of time. Starting value of $COMP_{ctrl}$ is 1. At CLK_{sample} rising edge $COMP_{ctrl}$ goes to 0 causing comparator to reset. When CLK_{sample} goes back to 0 $COMP_{ctrl}$ goes to one and first comparison of DAC's top plate voltages is performed. As a result (after comparison is finished) $VALID$ goes up which resets the comparator and in turn causes $VALID$ to go to 0 – this feedback loop is continued throughout whole conversion. CLK_0 is used to reset comparator at the end of conversion process. ACT_{comp} signal is additionally sensitive to RST_{DFF} to assure that comparator will be reset after RST_{DFF} goes to 1 (important in case of bad conversion).



(a) Simplified circuit schematic.

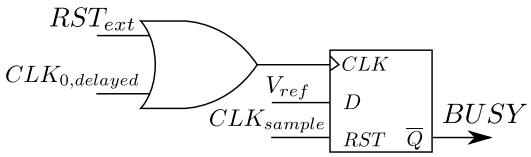


(b) Example of simulated waveforms.

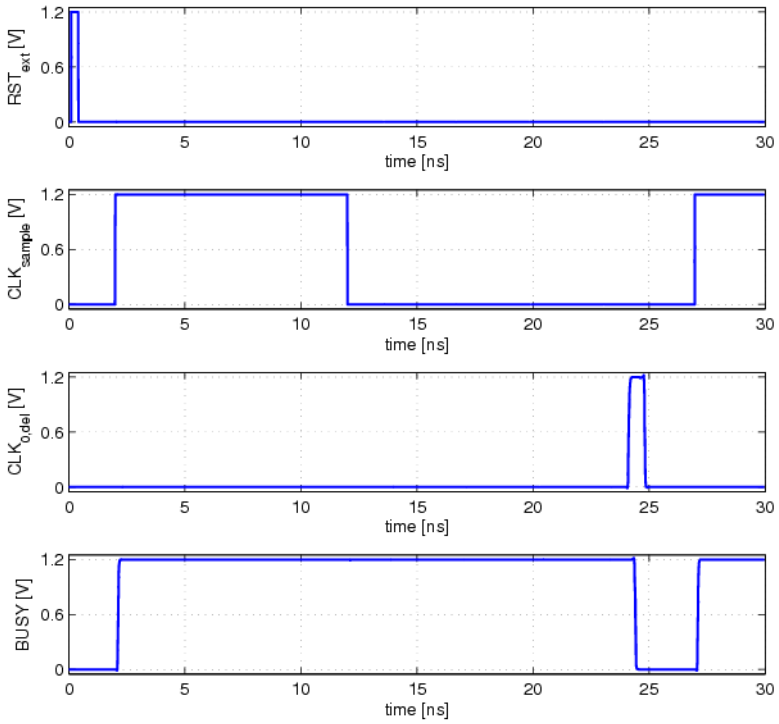
Figure 4.23: Comparator control circuit together with example of simulated waveforms.

BUSY generator

Task of this simple circuit is to generate $BUSY$ signal which would indicate if ADC is currently converting ($BUSY = 1$) or not ($BUSY = 0$). This functionality is obtained by using one DFF (static with reset, taken from IBM CMOS8RF library, which in contrast to all design DFFs is reset when RST signal is high) with input D connected to V_{ref}. At the start $BUSY$ is zero and goes high at the rising edge of CLK_{sample} (DFF is reset, so it's $\overline{Q}$ output goes to 1) and remains in this state until $CLK_{0,delayed}$ goes high which results in triggering DFF and $BUSY$ going to logical 0. In case of CLK_{sample} rising before end of current conversion $BUSY$ remains high.



(a) Simplified circuit schematic.

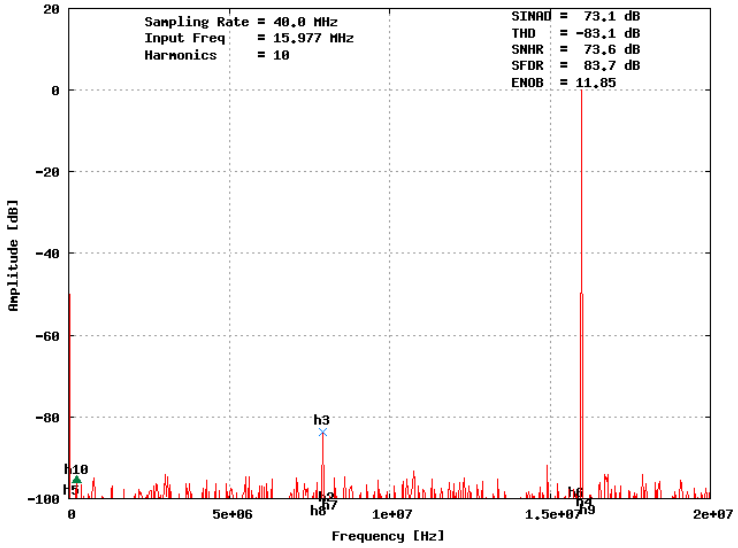


(b) Simulated waveforms.

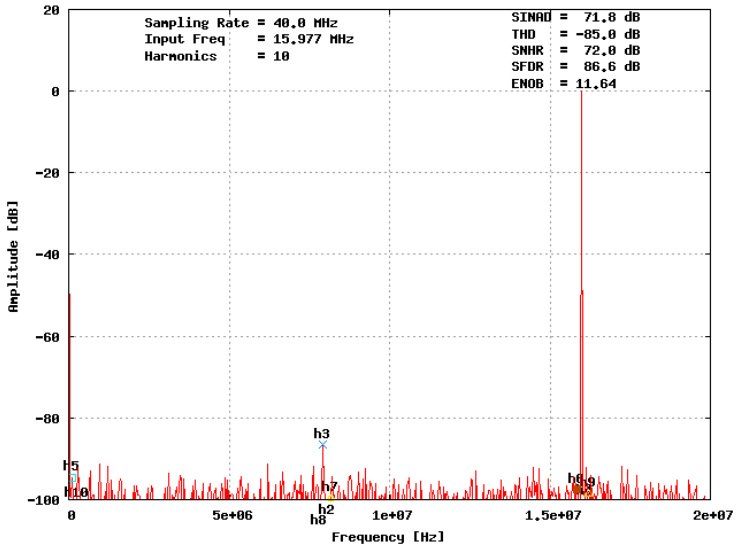
Figure 4.24: BUSY signal generator with simulated waveforms of circuit operations.

4.6 Performance of designed ADC

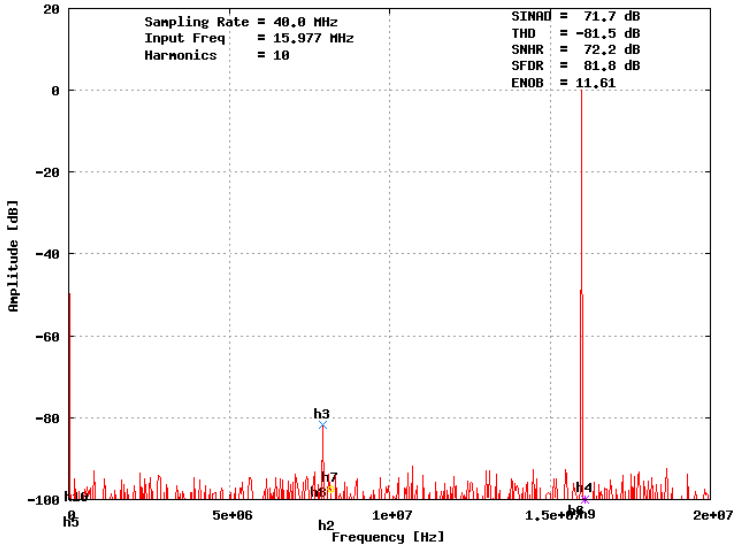
Result of Discrete Fourier Transforms obtained from simulation of schematics of designed ADC with MIM-capacitor based DACs are presented on Figure 4.25. Simulations were done for 40MS/s (signal frequency was chosen according to rules described in section 2.1.2, which turned out to be nearly 16MHz), for each DFT 1024 samples were gathered. Analogical simulation results with VerilogA models of functional blocks and MOM-based DAC extract were shown in Figure 4.7. Average power consumption for 40Ms/s conversion rate (for both analog power domain P_{ana} and digital power domain P_{dig}) for all ADCs is presented in Table 4.4.



(a) ADC with L4M7 DAC



(b) ADC with L4M7-0.5C DAC



(c) ADC with L3M8-0.5C DAC

Figure 4.25: Results of Discrete Fourier Transforms of simulations of full schematic of ADCs with different MIM-capacitor based DACs.

Table 4.4: Average power consumption of designed ADCs.

	P_{ana}@40MS/s [μW]	P_{dig}@40MS/s [μW]
MOM-based	306	326
MIM-based L4M7-0.5C	376	396
MIM-based L4M7	427	452
MIM-based L3M8-0.5C	397	421

Full schematic simulations show that designed ADCs work well – in two cases ENOB is about 11.6, while the best ADC (the one with L4M7 MIM-capacitor DAC) achieves 11.85 ENOB. Power consumption was kept below 1mW, which is well within design limits. Those are very good results for 12-bit ADC, but it must be remembered that next step in design would be physical layout and post-layout simulations, which would most likely show degradation in the performance due to parasitic capacitances and resistances.

Summary

The goal of this work was to design a 12-bit successive approximation analog-to-digital converter for possible use in future readout systems of High Energy Physics experiments. The converter should be able to work with 40MHz sampling clock, while maintaining very low power consumption and area. In the first part of the thesis overview of present and future HEP experiments was presented and the role of read-out microelectronics used in experimental devices was described. In chapter two basic definitions connected with ADCs were explained together with review of popular ADC architectures. Third chapter contains a detailed description of different approaches to successive approximation ADC and converter's building blocks. In the last chapter a detailed description of designed ADC illustrated with simulation results was presented.

In order to design a converter meeting required specifications a thorough examination of various approaches to successive approximation ADC was carried out, followed by development of Matlab code to calculate energy consumption of each configuration. This allowed to quantitatively compare all approaches in terms of power efficiency which, coupled with informations about other features of each method (e.g. number of needed voltage references, required internal DAC resolution, complexity of digital logic), lead to selection of Merged Capacitor Switching as the configuration implemented in the presented design.

Because focus of this work was put on research & development four different versions of ADC were designed (using Cadence software and IBM CMRF8SF 130nm technology) with the most important difference between them being the architecture of internal DAC (other functional blocks like bootstrapped sampling switches or DACs' switches and their buffers also differ between designed ADCs). Results of schematics-level simulations of ADCs showing their performance and total power consumption are presented in Table 4.5.

The results for ADCs with MIM-capacitor based DAC are quite pleasing – good dynamic performance, no missing code or major non-linear behaviour and power consumption kept well below 1mW.

Table 4.5: Summary of designed ADCs performance.

Used capacitors	SAR algorithm	ADC's internal DAC architecture	ENOB [bit]	INL_{max} [LSB]	DNL_{max} [LSB]	$P_{tot}@40MS/s$ [μW]
MIM	MCS	L4M7	11.85	0.65	0.55	879
		L4M7-0.5C	11.64	0.96	0.64	727
		L3M8	11.61	0.63	0.55	818
MOM	AMCS	11bit AMCS	11.57	2.74	1	818

It is worth stressing out that design of MOM-capacitor based DAC is first of its kind in Department of Particle Interactions and Detection Techniques WFiIS AGH and even though range of input signal's voltages is lower than in case of MIM-capacitor based DACs and much worse static performance is measured, using MOM-capacitors turned out to be a feasible way of significantly reducing area of the converter.

Future continuation of work presented in this thesis should start with making physical layouts of all remaining functional blocks. This will quite surely degrade performance of ADCs and might result in requirement for design modification e.g. adding calibration for DACs and comparator, improving design of sampling switches.

A | Energy cost of DAC switching for 3-bit classical ADC

Switching scheme for a 3-bit ADC using classical SAR algorithm is shown in Figure 3.2 and is repeated in this appendix in Figure A.1 for ease of reading. Power (energy) drawn from voltage supply V_{ref} due to capacitor switching during each transition can be calculated as follows (calculations is carried out for each capacitor separately based on equation 3.4, for better clarity notation E_{XY} will be used where X is size of capacitor in C_u and $Y = \{n, p\}$ indicates to which DAC capacitor belongs to):

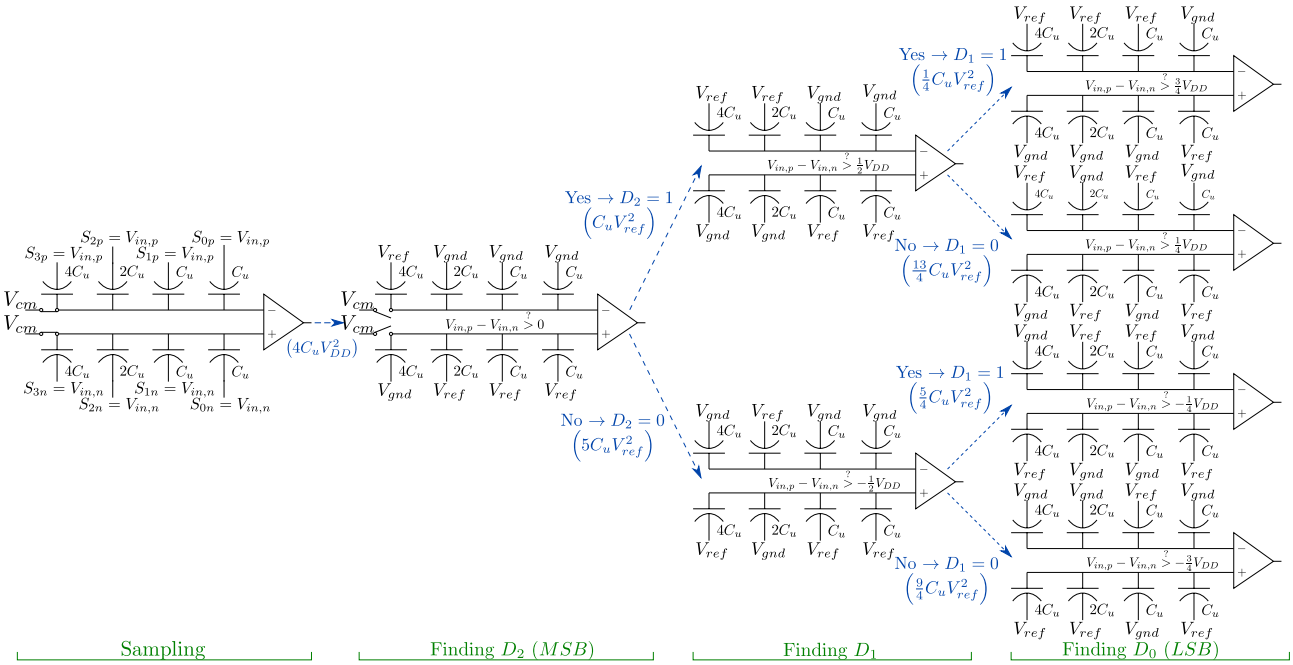


Figure A.1: 3-bit SAR ADC incorporating classical algorithm.

Switching after sampling

Top plate voltage of DAC sampling $V_{in,p}$ before switching $V_{tp,pre} = V_{cm}$

Top plate voltage of DAC sampling $V_{in,p}$ after switching $V_{tp,post} = V_{in,p} - V_{cm} + \frac{1}{2}V_{ref}$

Top plate voltage of DAC sampling $V_{in,n}$ before switching $V_{tn,pre} = V_{cm}$

Top plate voltage of DAC sampling $V_{in,n}$ after switching $V_{tn,post} = V_{in,n} - V_{cm} + \frac{1}{2}V_{ref}$

Energy consumption due to switching of $4C_u$ in DAC sampling $V_{in,p}$:

$E_{4p} = 4C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{in,p} - V_{tp,pre})\right] = 2C_uV_{ref}^2$

Energy consumption due to switching of $2C_u$ and two single C_u in DAC sampling $V_{in,n}$ (bottom plate voltages before and after switching for both capacitors are the same, so their energy consumption can be calculated simultaneously):

$E_{2n} + 2 \cdot E_{1n} = 4C_uV_{ref}\left[V_{ref} - V_{tn,post}\left(V_{in,n} - V_{tn,pre}\right)\right] = 2C_uV_{ref}^2$

Total energy consumption E_{tot} in this conversion step is algebraic sum of partial energies E_{4p}, E_{2n} and $2 \cdot E_{1n}$:

$E_{total} = E_{4p} + E_{2p} + 2 \cdot E_{1n} = 4C_uV_{ref}^2$

Switching after $D_2 = 1$

$V_{tp,pre} = V_{in,p} - V_{cm} + \frac{1}{2}V_{ref} \qquad V_{tp,post} = V_{in,p} - V_{cm} + \frac{3}{4}V_{ref}$

$V_{tn,pre} = V_{in,n} - V_{cm} + \frac{1}{2}V_{ref} \qquad V_{tn,post} = V_{in,n} - V_{cm} + \frac{1}{4}V_{ref}$

$E_{4p} = 4C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{ref} - V_{tp,pre})\right] = -C_uV_{ref}^2$

$E_{2p} = 2C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{gnd} - V_{tp,pre})\right] = \frac{3}{2}C_uV_{ref}^2$

$2 \cdot E_{1n} = 2C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{ref} - V_{tn,pre})\right] = \frac{1}{2}C_uV_{ref}^2$

$E_{total} = E_{4p} + E_{2p} + 2 \cdot E_{1n} = C_uV_{ref}^2$

Switching after $D_2 = 0$

$V_{tp,pre} = V_{in,p} - V_{cm} + \frac{1}{2}V_{ref} \qquad V_{tp,post} = V_{in,p} - V_{cm} + \frac{1}{4}V_{ref}$

$V_{tn,pre} = V_{in,n} - V_{cm} + \frac{1}{2}V_{ref} \qquad V_{tn,post} = V_{in,n} - V_{cm} + \frac{3}{4}V_{ref}$

$E_{2p} = 2C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{gnd} - V_{tp,pre})\right] = \frac{5}{2}C_uV_{ref}^2$

$E_{4n} = 4C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{gnd} - V_{tn,pre})\right] = 3C_uV_{ref}^2$

$2 \cdot E_{1n} = 2C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{ref} - V_{tn,pre})\right] = -\frac{1}{2}C_uV_{ref}^2$

$E_{total} = E_{2p} + E_{4n} + 2 \cdot E_{1n} = 5C_uV_{ref}^2$

Switching after $D_1 = 1$ (for $D_2 = 1$)

$$V_{tp,pre} = V_{in,p} - V_{cm} + \frac{3}{4}V_{ref} \qquad V_{tp,post} = V_{in,p} - V_{cm} + \frac{7}{8}V_{ref}$$

$$V_{tn,pre} = V_{in,n} - V_{cm} + \frac{1}{4}V_{ref} \qquad V_{tn,post} = V_{in,n} - V_{cm} + \frac{1}{8}V_{ref}$$

$$E_{4p} + E_{2p} = 6C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{ref} - V_{tp,pre})\right] = -\frac{3}{4}C_uV_{ref}^2$$

$$E_{1p} = C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{gnd} - V_{tp,pre})\right] = \frac{7}{8}C_uV_{ref}^2$$

$$E_{1n} = C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{ref} - V_{tn,pre})\right] = \frac{1}{8}C_uV_{ref}^2$$

$$E_{total} = E_{4p} + E_{2p} + E_{1p} + E_{1n} = \frac{1}{4}C_uV_{ref}^2$$

Switching after $D_1 = 0$ (for $D_2 = 1$)

$$V_{tp,pre} = V_{in,p} - V_{cm} + \frac{3}{4}V_{ref} \qquad V_{tp,post} = V_{in,p} - V_{cm} + \frac{5}{8}V_{ref}$$

$$V_{tn,pre} = V_{in,n} - V_{cm} + \frac{1}{4}V_{ref} \qquad V_{tn,post} = V_{in,n} - V_{cm} + \frac{3}{8}V_{ref}$$

$$E_{4p} = 4C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{ref} - V_{tp,pre})\right] = \frac{1}{2}C_uV_{ref}^2$$

$$E_{1p} = C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{gnd} - V_{tp,pre})\right] = \frac{9}{8}C_uV_{ref}^2$$

$$E_{2n} = 2C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{gnd} - V_{tn,pre})\right] = \frac{7}{4}C_uV_{ref}^2$$

$$E_{1n} = C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{ref} - V_{tn,pre})\right] = -\frac{1}{8}C_uV_{ref}^2$$

$$E_{total} = E_{4p} + E_{1p} + E_{2n} + E_{1n} = \frac{13}{4}C_uV_{ref}^2$$

Switching after $D_1 = 1$ (for $D_2 = 0$)

$$V_{tp,pre} = V_{in,p} - V_{cm} + \frac{1}{4}V_{ref} \qquad V_{tp,post} = V_{in,p} - V_{cm} + \frac{3}{8}V_{ref}$$

$$V_{tn,pre} = V_{in,n} - V_{cm} + \frac{3}{4}V_{ref} \qquad V_{tn,post} = V_{in,n} - V_{cm} + \frac{5}{8}V_{ref}$$

$$E_{2p} = 2C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{ref} - V_{tp,pre})\right] = -\frac{1}{4}C_uV_{ref}^2$$

$$E_{1p} = C_uV_{ref}\left[V_{ref} - V_{tp,post} - (V_{gnd} - V_{tp,pre})\right] = \frac{7}{8}C_uV_{ref}^2$$

$$E_{4n} + E_{1n} = 5C_uV_{ref}\left[V_{ref} - V_{tn,post} - (V_{ref} - V_{tn,pre})\right] = \frac{5}{8}C_uV_{ref}^2$$

$$E_{total} = E_{2p} + E_{1p} + E_{4n} + E_{1n} = \frac{5}{4}C_uV_{ref}^2$$

Switching after $D_1 = 0$ (for $D_2 = 0$)

$$V_{tp,pre} = V_{in,p} - V_{cm} + \frac{1}{4}V_{ref} \qquad V_{tp,post} = V_{in,p} - V_{cm} + \frac{1}{8}V_{ref}$$

$$V_{tn,pre} = V_{in,n} - V_{cm} + \frac{3}{4}V_{ref} \qquad V_{tn,post} = V_{in,n} - V_{cm} + \frac{7}{8}V_{ref}$$

$$E_{1p} = C_u V_{ref} \left[V_{ref} - V_{tp,post} - (V_{gnd} - V_{tp,pre})\right] = \frac{9}{8}C_u V_{ref}^2$$

$$E_{4n} + E_{1n} = 5C_u V_{ref} \left[V_{ref} - V_{tn,post} - (V_{ref} - V_{tn,pre})\right] = -\frac{5}{8}C_u V_{ref}^2$$

$$E_{2n} = 2C_u V_{ref} \left[V_{ref} - V_{tn,post} - (V_{gnd} - V_{tn,pre})\right] = \frac{7}{4}C_u V_{ref}^2$$

$$E_{total} = E_{1p} + E_{4n} + E_{2n} + E_{1n} = \frac{9}{4}C_u V_{ref}^2$$

B | Logical effort method

Logical effort is method of sizing CMOS logical gates and adjusting their number to obtain lowest possible delay along chain of gates. It is based on simple model of logic CMOS gates, presented in Figure B.1, in which delay is a result of charging and discharging capacitors through resistors. In this Figure C_{in} represents capacitance of transistor's gates connected to input. Voltage level on input decides whether output will be connected to positive power supply by pull-up resistive network R_{ui} (modelling conducting transistors as resistors) or to negative power supply by pull-down resistor network R_{di}. Output of every gate in this model is loaded with two capacitances: parasitic capacitance of gate's components C_{pi} and capacitive load that needs to be driven by analysed logic gate (in most cases it is input capacitance of next CMOS gate in chain).

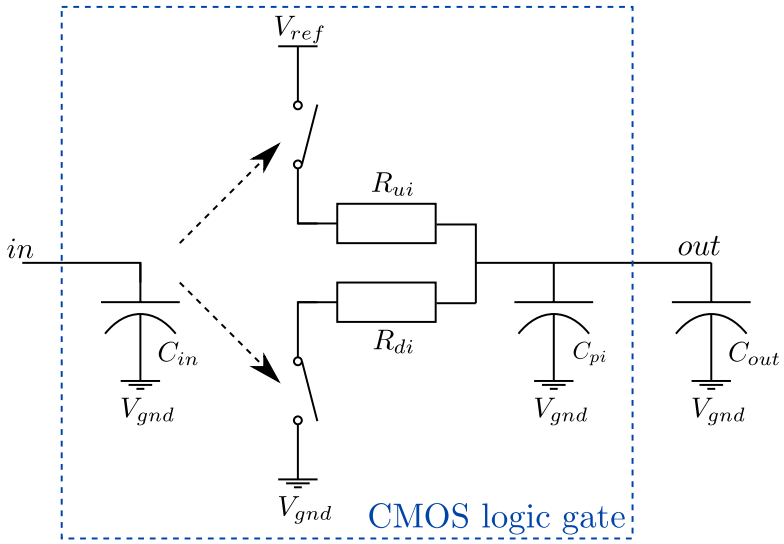


Figure B.1: Conceptual model of one input one output CMOS logical gate.

The main focus of described method is on minimizing delay by scaling size of transistors and/or changing number of gates in path. To make this approach easier every gate will be described as a scaled version of template circuit – to obtain given gate all transistor widths within it must be multiplied by scaling factor α. In simplest version of logical effort method additionally we assume that pull-up and pull-down resistances are equal. Four quantities characteristic for each gate ($C_{in}, C_{pi}, R_{di}, R_{ui}$) are related to their template equivalents (C_{ti}, C_{pt}, R_{ti}) by:

$$C_{in} = \alpha C_{ti} \tag{B.1}$$

$$C_{pi} = \alpha C_{pt} \tag{B.2}$$

$$R_{di} = R_{ui} = \frac{1}{\alpha} R_{ti} \tag{B.3}$$

In gate model as shown in Figure B.1 delay d_{abs} is a result of charging and discharging capacitor through resistor and can be calculated as (κ is fabrication process constant relating value of RC to delay in time):

$$d_{abs} = \kappa R_{ti} \left(C_{pt} + C_{out} \right) = \kappa R_{ti} C_{ti} \frac{C_{out}}{C_{in}} + \kappa R_{ti} C_{pt} \tag{B.4}$$

Scaling factor α is hidden within C_{in}.

Equation B.4 can be rewritten to obtain key equation of logical effort:

$$d_{abs} = \tau \left(gh + p \right) \tag{B.5}$$

Where equation components are defined as:

- $\tau = \kappa R_{inv} C_{inv}$ – so called delay unit, relating delay of given gate to that of inverter with logical effort of 1 and no parasitic delay (C_{inv} is inverter's input capacitance and R_{inv} is its pull-up or pull-down resistance). This value is characteristic for given fabrication process.
- $g = \frac{R_{ti} C_{ti}}{R_{inv} C_{inv}}$ – logical effort, relates RC constant of given gate to that of inverter (for inverter $g = 1$ is chosen). This value is determined by gate's topology.
- $h = \frac{C_{out}}{C_{in}}$ – electrical effort relates input and output capacitances and is only component of equation B.5 that is affected by scaling factor α (through value of C_{in})
- $p = \frac{R_{ti} C_{pt}}{R_{inv} C_{inv}}$ – parasitic delay, a fixed term (in relation to scale factor α) associated with gate topology

Total delay along path of N gates can be calculated as:

$$D = \sum_{i=1}^{N} \left(g_i h_i + p_i \right) \tag{B.6}$$

To calculate transistor size providing minimal path delay additional definitions are needed – total logical effort G and total electrical effort H are defined as:

$$G = \prod_{i=1}^{N} g_i \tag{B.7}$$

$$H = \frac{C_{out,N}}{C_{in,1}} \tag{B.8}$$

where $C_{in,1}$ is input capacitance of first gate in chain and $C_{out,N}$ is load capacitance for last gate.

To take into account that for branching paths only part of total current flows through branch along which calculations are carried out, a branching effort at the output of logic gate b is introduced:

$$b = \frac{C_{on-path} + C_{off-path}}{C_{off-path}} = \frac{C_{total}}{C_{off-path}} \tag{B.9}$$

where $C_{on-path}$ is load capacitance along analysed path while $C_{of-path}$ is load capacitance leading off anaylised path. Total branching effort B is defined as:

$$B = \prod_{i=1}^{N} b_i \tag{B.10}$$

Product of total branching effort and total electrical effort can be than calculated as:

$$BH = \frac{C_{out,N}}{C_{in,1}} \prod_{i=1}^{N} b_i = \prod_{i=1}^{N} h_i \tag{B.11}$$

Path effort F can therefore be calculated as:

$$F = GBH = \prod_{i=1}^{N} g_i h_i \tag{B.12}$$

Minimal path delay is obtained when all stages bear the same effort $\hat{f}$ (not necessarily implying the same delay per stage) [44]:

$$\hat{f} = \sqrt[N]{F} \tag{B.13}$$

Combining equations B.6 and B.13 results in expression for minimal delay along path $\hat{D}$:

$$\hat{D} = \sum_{i=1}^{N} \hat{f} + \sum_{i=1}^{N} p_i = N\hat{f} + P \tag{B.14}$$

After finding minimal delay sizes of transistors in logical gates that realize such performance can be calculated by working from last gate to the first one and for each of them calculating

its desired input capacitance $C_{in,i}$:

$$C_{in,i} = \frac{g_i}{\hat{f}} C_{out_i} \tag{B.15}$$

Logical effort for each gate can be calculated based on its definition – it is the ratio of gate's input capacitance to input capacitance of inverter with the same drive. Result of this calculation compared with equation B.1 provides needed scaling factor α for given CMOS gate. If for some reasons this ideal value of scale factor cannot be used (e.g. matching or restrictions in transistors width) the closest possible should be chosen – sizing obtained through logical effort method is quite flexible i.e. stages twice too small or twice too large result in delay only 15% worse than minimal [44].

Although logical effort method provides simple and efficient algorithm for sizing logical gates to obtain minimal delay, it has some shortcomings:

- modeling gate with just RC delay is overly simplistic – variable rise times of signals or effects such as velocity saturation in transistors are not taken into account
- only goal of this method is to obtain lowest delay – there are completely none power or area optimization
- when presented with complicated branching paths with large number of stages calculations become difficult

C | Matlab code for SAR algorithms energy consumption calculation

In this appendix Matlab scripts used to calculate DAC switching power consumption are presented. All function require the same input:

- N – resolution of ADC (in bits, must be integral number)
- input – input value of ADC (in LSB)

Output of all functions is also the same:

- en_vref – energy (in $C_u V_{ref}^2$) drawn during conversion from V_{ref}
- en_vcm – energy (in $C_u V_{ref}^2$) drawn during conversion from V_{cm}

Value of V_{cm} can be set within code itself (as fraction of V_{ref}). All calculations are based on equation 3.4.

Classical algorithm

```
function [en_vref en_vcm] = calc_energy_classic(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N+1);
caps(1,N+1) = 1;
for i=1:N
   caps(1,i) = 2^(N-i);
end
sp_pre = zeros(1,N+1);
sp_post = zeros(1,N+1);
sn_pre = zeros(1,N+1);
sn_post = zeros(1,N+1);
en_vref = 2^(N-1);
en_vcm = 0;
sp_pre(1,1) = 1;
sp_post(1,1) = 1;
sn_pre(1,1) = 0;
sn_post(1,1) = 0;
for i=2:(N+1)
   sp_pre(1,i) = 0;
   sp_post(1,i) = 0;
   sn_pre(1,i) = 1;
   sn_post(1,i) = 1;
end
vp_pre = 0.5;
vp_post = 0.5;
vn_pre = 0.5;
vn_post = 0.5;
for D=1:(N-1)
    if input_bin(1,D) == 1
        sp_post(1,D+1) = 1;
        sn_post(1,D+1) = 0;
    else
        sp_post(1,D+1) = 1;
        sp_post(1,D) = 0;
        sn_post(1,D+1) = 0;
        sn_post(1,D) = 1;
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps);
    for i=1:(N+1)
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

Energy saving

```
function [en_vref en_vcm] = calc_energy_energy_saving(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N-1);
caps(1,N-1) = 1;
for i=1:(N-2)
   caps(1,i) = 2^(N-i-2);
end
cap_msb = 2^(N-1);
caps_tot = 2^N;
spM_pre = 0;
spM_post = 0;
snM_pre = 0;
snM_post = 0;
spM2_pre = zeros(1,N-1);
spM2_post = zeros(1,N-1);
spL_pre = zeros(1,N-1);
spL_post = zeros(1,N-1);
snM2_pre = zeros(1,N-1);
snM2_post = zeros(1,N-1);
snL_pre = zeros(1,N-1);
snL_post = zeros(1,N-1);

en_vref = 0;
en_vcm = 0;

vp_pre = 1;
vp_post = 1;
vn_pre = 1;
vn_post = 1;

for D=1:(N-1)
    if D == 1
       if input_bin(1,D) == 1
           spM_post = 1;
           snM_post = 0;
           for i= 1:(N-1)
               spM2_post(1,i) = 0;
               snM2_post(1,i) = 1;
               spL_post(1,i) = 1;
               snL_post(1,i) = 0;
           end
       else
           spM_post = 0;
           snM_post = 1;
           for i= 1:(N-1)
               spM2_post(1,i) = 1;
               snM2_post(1,i) = 0;
               spL_post(1,i) = 0;
               snL_post(1,i) = 1;
           end
       end
    else
        if input_bin(1,1) == 1
            if input_bin(1,D) == 1
                spM2_post(1,D-1) = 1;
                snM2_post(1,D-1) = 0;
            else
                spL_post(1,D-1) = 0;
                snL_post(1,D-1) = 1;
            end
        else
            if input_bin(1,D) == 1
                spL_post(1,D-1) = 1;
                snL_post(1,D-1) = 0;
            else
                spM2_post(1,D-1) = 0;
                snM2_post(1,D-1) = 1;
            end
       end
    end

    vp_post = 1 + caps_to_vx(cap_msb, spM_post, 1)/sum(caps_tot) + caps_to_vx(caps, spM2_post, 1)/sum(caps_tot) + ...
          caps_to_vx(caps, spL_post, 1)/sum(caps_tot);
    vn_post = 1 + caps_to_vx(cap_msb, snM_post, 1)/sum(caps_tot) + caps_to_vx(caps, snM2_post, 1)/sum(caps_tot) + ...
          caps_to_vx(caps, snL_post, 1)/sum(caps_tot);

    if spM_post == 1
        en_vref = en_vref + cap_msb*(spM_post - vp_post - (spM_pre - vp_pre));
    end
    if snM_post == 1
            en_vref = en_vref + cap_msb*(snM_post - vn_post - (snM_pre - vn_pre));
    end
    for i=1:(N-1)
        if spM2_post(1,i) == 1
            en_vref = en_vref + caps(i)*(spM2_post(1,i) - vp_post - (spM2_pre(1,i) - vp_pre));
        end
        if spL_post(1,i) == 1
            en_vref = en_vref + caps(i)*(spL_post(1,i) - vp_post - (spL_pre(1,i) - vp_pre));
        end
        if snM2_post(1,i) == 1
            en_vref = en_vref + caps(i)*(snM2_post(1,i) - vn_post - (snM2_pre(1,i) - vn_pre));
        end
        if snL_post(1,i) == 1
            en_vref = en_vref + caps(i)*(snL_post(1,i) - vn_post - (snL_pre(1,i) - vn_pre));
        end
    end
    spM_pre = spM_post;
    spM2_pre = spM2_post;
    spL_pre = spL_post;
    snM_pre = snM_post;
    snM2_pre = snM2_post;
    snL_pre = snL_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

Monotonic

```
function [en_vref en_vcm] = calc_energy_monotonic(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N);
caps(1,N) = 1;
for i=1:(N-1)
   caps(1,i) = 2^(N-i-1);
end
sp_pre = zeros(1,N);
sp_post = zeros(1,N);
```

```
sn_pre = zeros(1,N);
sn_post = zeros(1,N);
en_vref = 0;
en_vcm = 0;

for i=1:N
   sp_pre(1,i) = 1;
   sp_post(1,i) = 1;
   sn_pre(1,i) = 1;
   sn_post(1,i) = 1;
end
vp_pre = 1;
vp_post = 1;
vn_pre = 1;
vn_post = 1;
for D=1:(N-1)
    if input_bin(1,D) == 1
        sp_post(1,D) = 0;
    else
        sn_post(1,D) = 0;
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps);
    for i=1:N
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

MCS

```
function [en_vref en_vcm] = calc_energy_mcs(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N);
caps(1,N) = 1;
for i=1:(N-1)
   caps(1,i) = 2^(N-i-1);
end
sp_pre = zeros(1,N);
sp_post = zeros(1,N);
sn_pre = zeros(1,N);
sn_post = zeros(1,N);
en_vref = 0;
en_vcm = 0;
vcm = 0.5;  % what fraction of Vref is Vcm

for i=1:N
   sp_pre(1,i) = vcm;
   sp_post(1,i) = vcm;
   sn_pre(1,i) = vcm;
   sn_post(1,i) = vcm;
end
vp_pre = vcm;
vp_post = vcm;
vn_pre = vcm;
vn_post = vcm;
for D=1:(N-1)
    if input_bin(1,D) == 1
        sp_post(1,D) = 0;
        sn_post(1,D) = 1;
    else
        sp_post(1,D) = 1;
        sn_post(1,D) = 0;
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
    for i=1:N
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sp_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
        if sn_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

EMCS

```
function [en_vref en_vcm] = calc_energy_emcs(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N);
caps(1,N) = 1;
for i=1:(N-1)
   caps(1,i) = 2^(N-i-1);
end
sp_pre = zeros(1,N);
sp_post = zeros(1,N);
sn_pre = zeros(1,N);
sn_post = zeros(1,N);
en_vref = 0;
en_vcm = 0;
vcm = 0.5;  % what fraction of Vref is Vcm

for i=1:N
   sp_pre(1,i) = vcm;
   sp_post(1,i) = vcm;
   sn_pre(1,i) = vcm;
```

```
    sn_post(1,i) = vcm;
end
vp_pre = vcm;
vp_post = vcm;
vn_pre = vcm;
vn_post = vcm;
for D=1:(N-1)
    if D == 1
        if input_bin(1,D) == 1
            sp_post(1,D) = 0;
            sn_post(1,D) = 1;
        else
            sp_post(1,D) = 1;
            sn_post(1,D) = 0;
        end

    else
        if input_bin(1,D) == sn_pre(1,D-1)
            if input_bin(1,D) == 1
                sp_post(1,D) = 0;
                sn_post(1,D) = 1;
            else
                sp_post(1,D) = 1;
                sn_post(1,D) = 0;
            end
        else
            if input_bin(1,D) == 1
                sp_post(1,D-1) = vcm;
                sn_post(1,D-1) = vcm;

                vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
                vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
                for i=1:N
                    if sp_post(1,i) == 1
                        en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
                    end
                    if sp_post(1,i) == vcm
                        en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
                    end
                    if sn_post(1,i) == 1
                        en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
                    end
                    if sn_post(1,i) == vcm
                        en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
                    end
                end
                sp_pre = sp_post; sn_pre = sn_post; vp_pre = vp_post; vn_pre = vn_post;
                sp_post(1,D) = 1;
                sn_post(1,D) = 0;
            else
                sp_post(1,D-1) = vcm;
                sn_post(1,D-1) = vcm;
                vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
                vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
                for i=1:N
                    if sp_post(1,i) == 1
                        en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
                    end
                    if sp_post(1,i) == vcm
                        en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
                    end
                    if sn_post(1,i) == 1
                        en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
                    end
                    if sn_post(1,i) == vcm
                        en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
                    end
                end
                sp_pre = sp_post; sn_pre = sn_post; vp_pre = vp_post; vn_pre = vn_post;
                sp_post(1,D) = 0;
                sn_post(1,D) = 1;

            end
        end
    end
        vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
        vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
        for i=1:N
            if sp_post(1,i) == 1
                en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
            end
            if sp_post(1,i) == vcm
                en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
            end
            if sn_post(1,i) == 1
                en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
            end
            if sn_post(1,i) == vcm
                en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
            end
        end
        sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

AMCS

```
function [en_vref en_vcm] = calc_energy_amcs(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N-1);
caps(1,N-1) = 1;
for i=1:(N-2)
   caps(1,i) = 2^(N-i-2);
end
sp_pre = zeros(1,N-1);
sp_post = zeros(1,N-1);
sn_pre = zeros(1,N-1);
sn_post = zeros(1,N-1);
en_vref = 0;
en_vcm = 0;
vcm = 0.5;  % what fraction of Vref is Vcm
for i=1:(N-1)
   sp_pre(1,i) = vcm;
   sp_post(1,i) = vcm;
   sn_pre(1,i) = vcm;
```

```
    sn_post(1,i) = vcm;
end
vp_pre = vcm;
vp_post = vcm;
vn_pre = vcm;
vn_post = vcm;
for D=1:(N-1)
    if D ~= (N-1)
        if input_bin(1,D) == 1
            sp_post(1,D) = 0;
            sn_post(1,D) = 1;
        else
            sp_post(1,D) = 1;
            sn_post(1,D) = 0;
        end
    else
       if input_bin(1,D) == 1
            sp_post(1,D) = 0;
            sn_post(1,D) = vcm;
        else
            sp_post(1,D) = vcm;
            sn_post(1,D) = 0;
        end
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
    for i=1:(N-1)
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sp_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
        if sn_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

Tri-level switching

```
function [en_vref en_vcm] = calc_energy_trilevel(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N-1);
caps(1,N-1) = 1;
for i=1:(N-2)
   caps(1,i) = 2^(N-i-2);
end
sp_pre = zeros(1,N-1);
sp_post = zeros(1,N-1);
sn_pre = zeros(1,N-1);
sn_post = zeros(1,N-1);

en_vref = 0;
en_vcm = 0;
vcm = 0.5;  % what fraction of Vref is Vcm

for i=1:(N-1)
   sp_pre(1,i) = 0;
   sp_post(1,i) = 0;
   sn_pre(1,i) = 0;
   sn_post(1,i) = 0;
end
vp_pre = 0;
vp_post = 0;
vn_pre = 0;
vn_post = 0;

sw_side = 0;       % 0 - switching pDAC; 1 - switching nDAC;

for D=1:(N-1)
    if D == 1
        if input_bin(1,D) == 1
            for i = 1:(N-1)
                sp_post(1,i) = 0;
                sn_post(1,i) = vcm;
                sw_side = 1;
            end
        else
            for i = 1:(N-1)
                sp_post(1,i) = vcm;
                sn_post(1,i) = 0;
                sw_side = 0;
            end
        end
    else
        if sw_side == 0
            if input_bin(1,D) == 1
                sp_post(1,D-1) = 0;
            else
                sp_post(1,D-1) = 1;
            end
        end
        if sw_side == 1
            if input_bin(1,D) == 1
                sn_post(1,D-1) = 1;
            else
                sn_post(1,D-1) = 0;
            end
        end
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
    for i=1:(N-1)
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sp_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
```

```
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
        if sn_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

Switchback

```
function [en_vref en_vcm] = calc_energy_switchback(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N);
caps(1,N) = 1;
for i=1:(N-1)
   caps(1,i) = 2^(N-i-1);
end
sp_pre = zeros(1,N);
sp_post = zeros(1,N);
sn_pre = zeros(1,N);
sn_post = zeros(1,N);
sp_pre(1,1) = 0;
sp_post(1,1) = 0;
sn_pre(1,1) = 0;
sn_post(1,1) = 0;
for i=2:N
   sp_pre(1,i) = 1;
   sp_post(1,i) = 1;
   sn_pre(1,i) = 1;
   sn_post(1,i) = 1;
end
vp_pre = 0.5;
vp_post = 0.5;
vn_pre = 0.5;
vn_post = 0.5;

en_vref = 2^(N-2);
en_vcm = 0;

for D=1:(N-1)
    if D == 1
        if input_bin(1,D) == 1
            sp_post(1,D) = 0;
            sn_post(1,D) = 1;
        else
            sp_post(1,D) = 1;
            sn_post(1,D) = 0;
        end
    else
        if input_bin(1,D) == 1
            sp_post(1,D) = 0;
        else
            sn_post(1,D) = 0;
        end
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps);
    for i=1:N
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

Improved switchback

```
function [en_vref en_vcm] = calc_energy_imp_switchback(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N-1);
caps(1,N-1) = 1;
for i=1:(N-2)
   caps(1,i) = 2^(N-i-2);
end
sp_pre = zeros(1,N-1);
sp_post = zeros(1,N-1);
sn_pre = zeros(1,N-1);
sn_post = zeros(1,N-1);
sp_pre(1,1) = 0;
sp_post(1,1) = 0;
sn_pre(1,1) = 0;
sn_post(1,1) = 0;
for i=2:(N-1)
   sp_pre(1,i) = 1;
   sp_post(1,i) = 1;
   sn_pre(1,i) = 1;
   sn_post(1,i) = 1;
end
vp_pre = 0.5;
vp_post = 0.5;
vn_pre = 0.5;
vn_post = 0.5;

en_vref = 2^(N-3);
en_vcm = 0;
vcm = 0.5;  % what fraction of Vref is Vcm

for D=1:(N-1)
    if D == 1
        if input_bin(1,D) == 1
            sp_post(1,D) = 0;
            sn_post(1,D) = 1;
        else
            sp_post(1,D) = 1;
```

```
            sn_post(1,D) = 0;
        end
    elseif D == N-1
        if input_bin(1,D) == 1
            sp_post(1,D) = vcm;
            sn_post(1,D) = 1;
        else
            sp_post(1,D) = 1;
            sn_post(1,D) =vcm;
        end
    else
        if input_bin(1,D) == 1
            if sp_pre(1,D-1) == 1
                sp_post(1,D-1) = vcm;
            else
                sp_post(1,D) = 0;
            end
        else
            if sn_pre(1,D-1) == 1
                sn_post(1,D-1) = vcm;
            else
                sn_post(1,D) = 0;
            end
        end
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
    for i=1:(N-1)
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sp_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
        if sn_post(1,i) == vcm
            en_vcm = en_vcm + 0.5*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
    end
    sp_pre = sp_post;
    sn_pre = sn_post;
    vp_pre = vp_post;
    vn_pre = vn_post;
end
```

V_{cm}-based monotonic

```
function [en_vref en_vcm] = calc_energy_vcm_monotonic(N, input)
input_bin = dec2bin(input,N) - '0';
caps = zeros(1,N-1);
caps(1,N-1) = 1;
for i=1:(N-2)
   caps(1,i) = 2^(N-i-2);
end
sp_pre = zeros(1,N-1);
sp_post = zeros(1,N-1);
sn_pre = zeros(1,N-1);
sn_post = zeros(1,N-1);

en_vref = 0;
en_vcm = 0;
vcm = 0.5;  % what fraction of Vref is Vcm

for i=1:(N-1)
   sp_pre(1,i) = vcm;
   sp_post(1,i) = vcm;
   sn_pre(1,i) = vcm;
   sn_post(1,i) = vcm;
end
vp_pre = vcm;
vp_post = vcm;
vn_pre = vcm;
vn_post = vcm;

for D=1:(N-1)
    if D == 1
        if input_bin(1,D) == 1
            for i = 1:(N-1)
                sp_post(1,i) = vcm;
                sn_post(1,i) = 1;
            end
        else
            for i = 1:(N-1)
                sp_post(1,i) = 1;
                sn_post(1,i) = vcm;
            end
        end
    else
        if input_bin(1,1) == 1
            if input_bin(1,D) == 1
                sp_post(1,D-1) = 0;
            else
                sn_post(1,D-1) = 0.5;
            end
        end
        if input_bin(1,1) == 0
            if input_bin(1,D) == 1
                sp_post(1,D-1) = 0.5;
            else
                sn_post(1,D-1) = 0;
            end
        end
    end
    vp_post = caps_to_vx(caps, sp_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sp_post, vcm)/sum(caps);
    vn_post = caps_to_vx(caps, sn_post, 1)/sum(caps) + vcm*caps_to_vx(caps, sn_post, vcm)/sum(caps);
    for i=1:(N-1)
        if sp_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sp_post(1,i) == vcm
            en_vcm = en_vcm + vcm*caps(i)*(sp_post(1,i) - vp_post - (sp_pre(1,i) - vp_pre));
        end
        if sn_post(1,i) == 1
            en_vref = en_vref + caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
        end
```

```
69          if sn_post(1,i) == vcm
70              en_vcm = en_vcm + vcm*caps(i)*(sn_post(1,i) - vn_post - (sn_pre(1,i) - vn_pre));
71          end
72      end
73      sp_pre = sp_post;
74      sn_pre = sn_post;
75      vp_pre = vp_post;
76      vn_pre = vn_post;
77  end
```

List of Figures

List of Tables

Bibliography

[1] ILC Collaboration, *The International Linear Collider*, http://www.linearcollider.org, 2007.

[2] CERN Press Release, *CERN experiments observe particle consistent with long-sought Higgs boson*, Geneve, 4^{th} July 2012.

[3] CERN Press Release, *New results indicate that particle discovered at CERN is a Higgs boson*, Geneve, 14^{th} March 2012.

[4] K. Schindl, *The Injector Chain for the LHC*, CERN-OPEN-99-052.

[5] CERN Document Service, http://cds.cern.ch, 2013

[6] Sz. Kulis, *Development of prototype luminosity detector modules for future experiments on linear colliders*, AGH University of Science and Technology, Kraków, 2012.

[7] Meenakshi Narain *Where is the New Physics? Results from LHC* International Workshop on Future Linear Colliders, October 2012

[8] S.-H. Cho, Ch.-K. Lee, J.-K. Kwon, S.-T. Ryu, *A 550-μW 10-b 40-MS/s SAR ADC With Multistep Addition-Only Digital Error Correction*, IEEE Journal of Solid State Circuits, vol. 46, no. 8, August 2011.

[9] M. J. M. Pelgrom, *Analog-to-Digital Conversion*, Second edition, Springer, Netherlands 2012.

[10] Sz. Kulis, *Projekt elektroniki front-end dla kalorymetru LumiCal dla ILC*, AGH University of Science and Technology, Kraków, 2008.

[11] R. van der Plassche, *CMOS integrated analog-to-digital and digital-to-analog converters*, Kluwer academic publisher, 2003.

[12] R. Jacob Baker, *CMOS Circuit Design, Layout and Simulation*, 3^{rd} edition Wiley, 2010.

[13] J. Holub, J. Vedral, *Stochastic testing of ADC Step-Gauss method*, Computer Standards & Interfaces, Volume 26, Issue 3, 2004.

[14] F. Maloberti, *Data converters*, Springer, Netherlands 2004.

[15] R. G. Lyons, *Understanding Digital Signal Processing*, Adison Wesley Longman Inc., 1997.

[16] IEEE-SA Standards Board, *IEEE Standard for Terminology and Test Methods for Analog-to-Digital Converters (IEEE Std 1241-2010)*, IEEE Instrumentation and Measurement Society, January 2011.

[17] W. Kester, *Which ADC Architecture is right for your application?*, Analog Dialogue, vol. 39, no. 6, June 2005.

[18] B.P. Ginsburg, A. P. Chandrakasan, *An Energy-Efficient Charge Recycling Approach for a SAR Converter With Capacitive DAC*, IEEE ISCAS, vol. 1, May 2005.

[19] Ch.-Ch. Liu, S.-J. Chang, G.-Y. Huang, Y.-Z. Lin, *A 10-bit 50-MS/s SAR ADC With a Monotonic Capacitor Switching Procedure*, IEEE Journal of Solid State Circuits, vol. 45, no. 4, April 2010.

[20] Y.-K. Chang, Ch.-S. Wang, Ch.-K. Wang, *A 8-bit 500-KS/s Low Power SAR ADC for Bio-Medical Application*, IEEE Asian Solid-State Circuits Conferance, November 2007.

[21] Y. Zhu, Ch,-H Chan, U-F. Chio, S.-W. Sin, S.-P. U, R.-P. Martins, F. Maloberti, *A 10-bit 100-MS/s Reference-Free SAR ADC in 90 nm CMOS*, IEEE Journal of Solid State Circuits, vol. 45, no. 6, June 2010.

[22] V. Hariprasath, J. Guerber, S.-H. Lee, U.-K. Moon, *Merged capacitor switching based SAR ADC with highest switching energy-efficiency*, Electronics Letters, vol. 46, no. 9, April 2010.

[23] J. Guerber, H. Venkatram, T. Oh, U.-K. Moon, *Enhanced SAR ADC Energy Efficiency from the Early Reset Merged Capacitor Switching Algorithm*, IEEE International Symposium on Circuits and Systems, May 2012.

[24] Y.-K. Cho., Y.-D. Jeon, J.-W. Nam, J.K. Kwon, *A 9-bit 80 MS/s Successive Approximation Register Analog-to-Digital Converter With a Capacitor Reduction Technique*, IEEE Transactions on Circuits and Sytems, vol. 57, no. 10, July 2010.

[25] B. Sedighi, A.T. Huynh, E. Skafidas, *Design of the internal DAC in SAR ADCs*, IEEE International Midwest Symposium on Circuits and Systems, August 2012.

[26] C. Yuan, Y. Lam, *Low-energy and area-efficient tri-level switching scheme for SAR ADC*, Electronics Letters, vol. 48, no. 9, April 2012.

[27] G.-Y. Huang, S.-J. Chang, Ch.-Ch. Liu, Y.-Z. Lin, *10-bit 30MS/s SAR ADC Using a Switchback Switching Method*, IEEE Transactions on VLSI Systems, vol. 21, no. 3, March 2013.

[28] A. Sanyal, N.Sun, *SAR ADC architecture with 98% recution in switching energy over convertional scheme*, Electronics Letters, vol. 49, no. 4, February 2013.

[29] Z. Zhu, Y. Xiao, X. Song, *V_{cm}-based monotonic capacitor switching scheme for SAR ADC*, Electronics Letters, vol. 49, no. 5, February 2013.

[30] D. Zhang, *Design of Ultra-Low-Power Analog-to-Digital Converters*, Linköping University, Linköping, 2012.

[31] IBM Microelectronics Division – Mixed Signal Technology Development, *CMOS8RF(CMRF8SF) Design Manual*, November 2010.

[32] M. Saberi, R. Lofti, K. Mafinezhad, W. A. Serdijn, *Analysis of Power Consumption and Linearity in Capacitive Digital-to-Analog Converters Used in Successive Approximation ADCs*, IEEE Transactions on Circuits and Systems, vol. 58, no. 8, August 2011.

[33] A. Abusleme, A. Dragone, G. Haller and B. Murmann, *Mismatch of lateral field metal-oxide-metal capacitors in 180 nm CMOS process*, Electronics Letters, vol. 48, no. 5, March 2012.

[34] M. Waltari, *Circuit techniques for low-voltage and high-speed A/D converters*, Helsinki University of Technology, Helsinki, 2002.

[35] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw-Hill Education, 2002.

[36] M. Dessouky, A. Kaiser, *Input switch configuration suitable for rail-to-rail operation of switched opamp circuits*, Electronics Letters, vol. 35, no. 1, January 1999.

[37] H.J. Jeon, *Low-power high-speed low-offset fully dynamic CMOS latched comparator*, Electrical and Computer Engineering Master's Theses, Northeastern University, 2010.

[38] T. Cho, P. Grey, *A 10 b, 20 Msample/s, 35 mW pipeline A/D converter*, IEEE Journal of Solid-State Circuits, vol. 30, no. 3, March 1995.

[39] D. Schinkel, E. Mensink, E. Kiumperink, E. van Tuijl and B. Nauta, *A Double-Tail Latch-Type Voltage Sense Amplifier with 18ps Setup+Hold Time*, ISSCC Dig. Tech. Papers, February 2007.

[40] P. M. Gigueiredo, J. C. Vital, *Kickback Noise Reduction Techniques for CMOS Latched Comparators*, IEEE Transactions on Circuits and Systems-II: Express Briefs, vol. 53, no. 7, July 2006.

[41] J.E. Eklund, Ch. Svennson *Influence of Metastability Errors on SNR in Successive-Approximation A/D Converters*, Analog Integrated Circuits and Signal Processing, vol. 26, Kluwer Academic Publishers, 2001.

[42] A. Nikoozadeh, B. Murmann, *An Analysis of Latch Comparator Offset Due to Load Capacitor Mismatch*, IEEE Transactions on Circuits and Systems-II: Express Briefs, vol. 53, no. 12, December 2006.

[43] T.D. Low, *Low Power, High Performance Pseudo-static D Flip-Flop*, Oregon State University, 2000.

[44] I. Sutherland, B. Sproull, D. Harris, *Logical Effort: Designing Fast CMOS Circuits*, Morgan Kaufmann Publishers Inc., San Francisco, 1999.