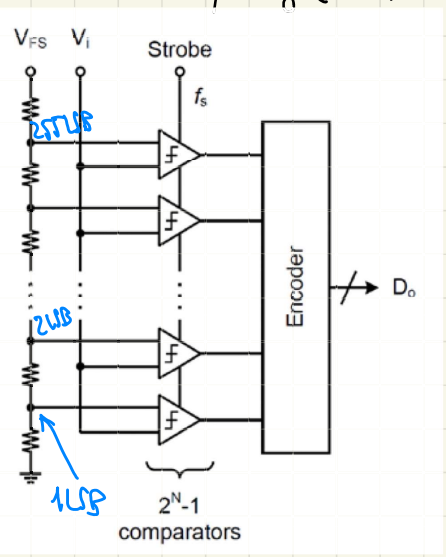
ES14 - ADVANCED ADCs

30/11/2021

Goals: – reducing T_{conv} (increase speed) } { serial (pipeline) processing
– reducing # of components } { parallel processing
↳ which means reducing Area & PC

SUBRANGING ADC

IDEA: In the flash ADC we create "all" possible voltage values from V_{ref} to V_{low} using resistors (2^N)
Then, using (2^N-1) comparators, we compare A_{in} to all these just created values

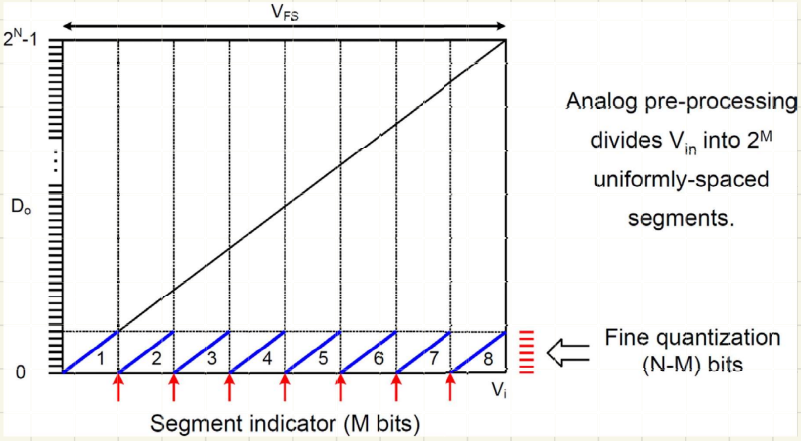


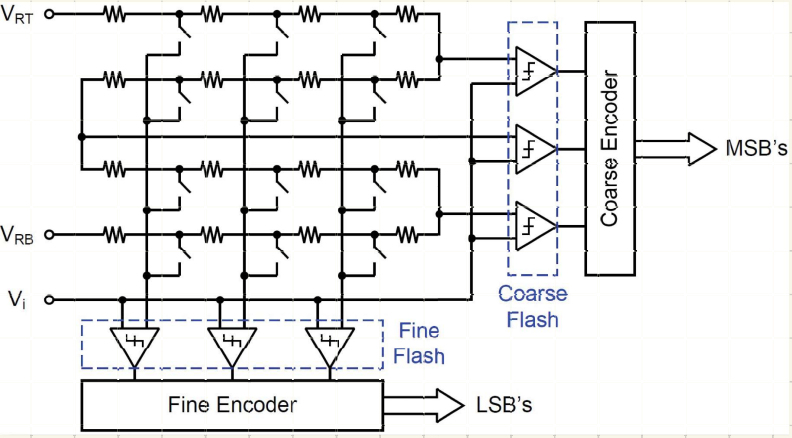
From a certain point downwards the comparators will be tripped bc V_{in} is lower than their respective V_{ref} while the previous comparators will not be bc V_{in} is lower than their respective V_{ref}
Then, changing V_{in}, some comparators will start tripping and others will stop tripping

Let's suppose to have $V_{in} = 2.5V = FSR/2 = V_{ref}/2$ so we'll have 1/2 comparators that are already tripped and 1/2 comparators that are not tripped yet
↓
The comparators that matter are those ones that are very close to V_{in}, bc they are those ones on which the encoder relies to decide the output code D_{out}
↓
The useless 1/2 comparators that are already tripped continue to dissipate power → ISSUE: POWER CONSUMPTION ☹

SOLUTION: SEGMENTED QUANTIZATION

basic idea: Let's use a lower # of comparators (i.e. 16 instead of 256) in such a way that they will give us a coarse quantization so they will give us just the MSB of our conversion and then once we decide where we are in the conv., we'll turn on the other 16 comparators, in such a way in each sub-range we have 16 LSBs, so eventually our 256 LSBs can be split in 16 macro-levels and each macro-level will decide which subset of comparators have to be turned on in order to complete the fine quantization of our analog input voltage A_{in}





At the beginning the switches are all open and only the coarse flash comparators are connected to the ladder.
Once the coarse comparators provide the coarse conversion, so once the coarse encoder understands where the Vin approximatively is (MSB conversion).

Only after the MSB conversion the coarse encoder will sequentially enable the switches: one is closed, while the others remain open
As a consequence, the fine comparators will turn on providing the fine conversion: the fine encoder understands where Vin exactly is!!!

We subdivided our FSR in, say, 16 levels and once we provide those 4 MSBs we understand in which of those 15 levels we are and so we turn on the remaining 16 switches in that specific macro-block (the macro-block depends on the result of the MSB conversion)

ADVANTAGE: we do not need 256 comparators but for an 8 bit ADC we need 4 bits for the MSB (coarse) conversion and 4 bits for the LSB (fine) conversion.
→ So in total we need 16 comparators for the MSB conv. (actually 15 bc7 one is always connected to the lowest possible value) and 16 comparators for to convert the LSBs

⇒ we only need 32 comparators instead of 256!! ☺

What does it mean?
- Area drastically reduced!
- Power consumption drastically reduced!

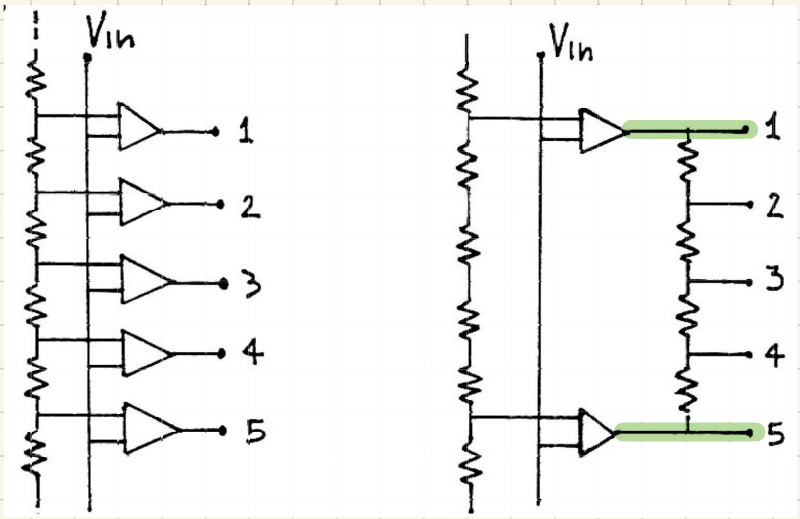
REMAINING ISSUES: • we still need 256 resistors

MAYBE NOT!??
- compared to a standard flash ADC, this subranging ADC takes almost 3 times to perform the conversion
  - $T_{conv,subranging} \simeq 3\, T_{conv,flash}$

The 16 fine comparators should have $V_{os,fine} \ll 1LSB = 19mV$ (it should be $V_{os,fine} \leq 1mV$)

INTERPOLATING FLASH ADC

Let's remove 3 comparators from every 4 comparators:



↓

This means that for a 8 bit ADC we'll have 64 comparators instead of 256

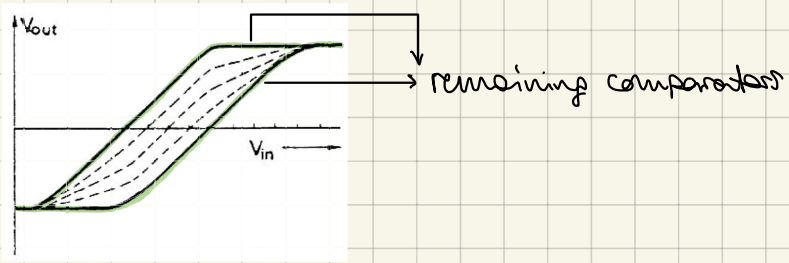
↓

Only 64 lines will reach the encoder that will produce a 6 bit ($2^6 = 64$) code

Now, let's add resistors btw our comparator output and the following comparator output

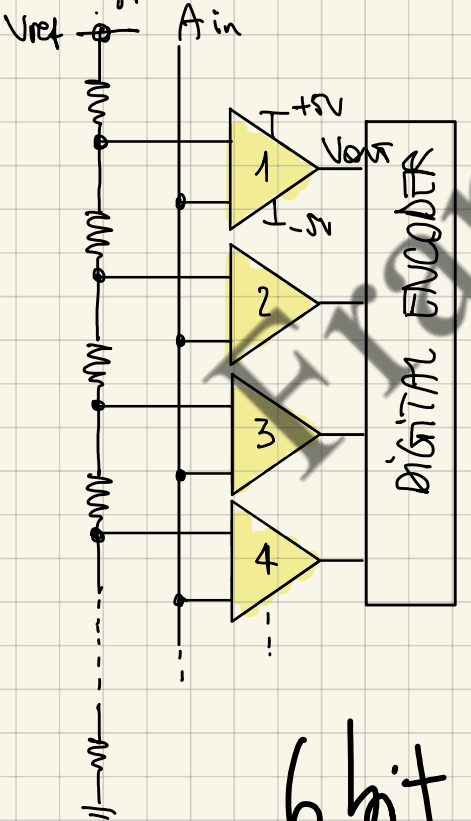
These added resistors will mimic the comparators we have removed

→ The 3 missing outputs are reconstructed w/ a network of resistor btw the remaining outputs



Vout is an index of similarity btw Vin and Vref

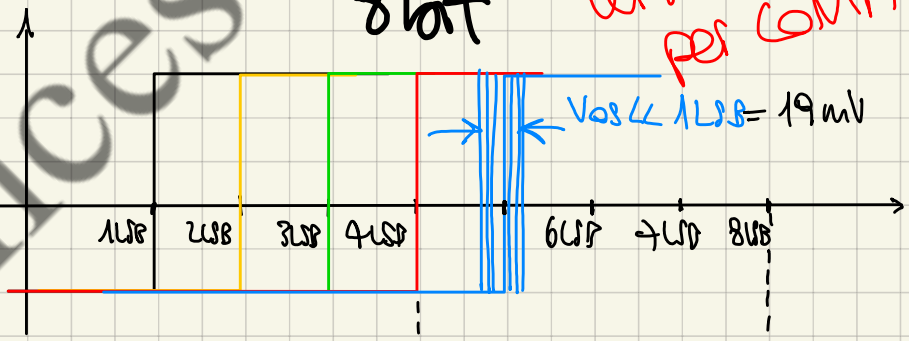
In a typical Flash ADC we have:



6 bit

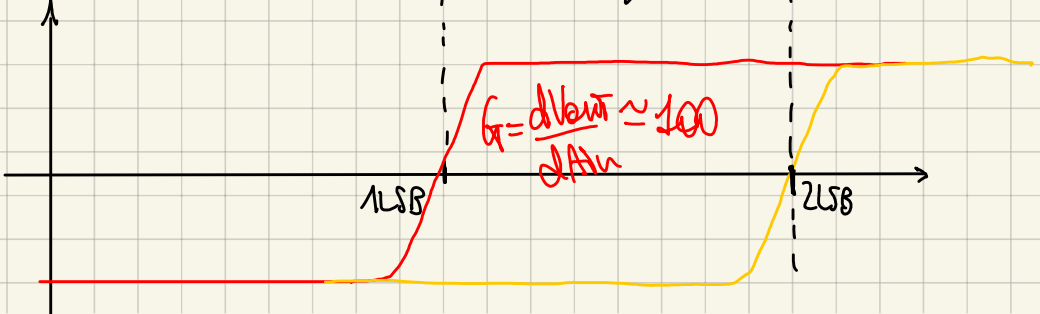
8 bit

let's 25 MOSFETS per COMPARATOR

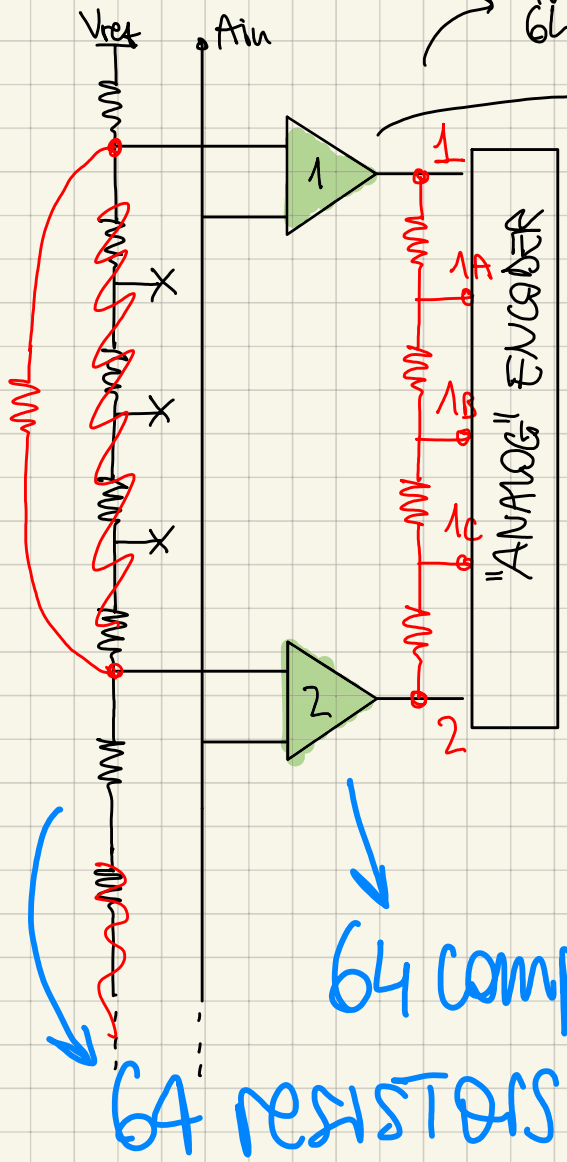


$\frac{dV_{out}}{dA_{in}} = \frac{10V}{(<<1LSB)} \simeq \frac{10V}{1mV} \simeq 10k$ → very high G comparator

This means that in a Flash ADC the higher the # of bits and so the # of comparators, the higher the # of resistor we must put inside each comparator to get a very very high gain

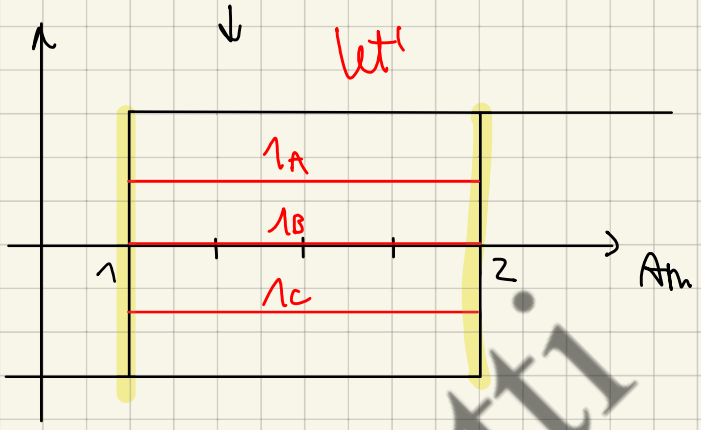


Now the idea is:

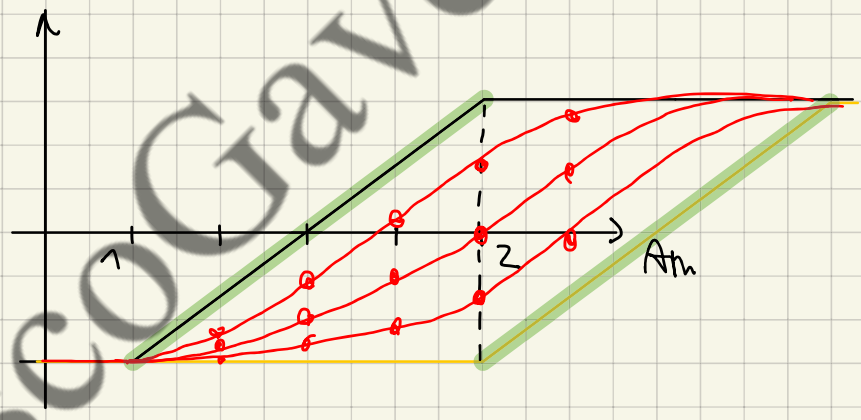


This behaves as a 6 bit ADC since it implements 64 comparators

if we use the same comparators as before we end up w/ an error



Let's use much purer comparators:



$$G = \frac{d\text{bit}}{dA_{in}} = \frac{10V}{1LSB_{6bitADC}} = \frac{10V}{5V/2^6} = \frac{10V}{\frac{5V}{2^6}\cdot 4} = \frac{1}{4} G_{8bit}$$

Now we have the same # of outputs and what we have to do is simply check if one output commutes or not.
In a flash ADC, when one comparator commutes, the output was either +5V or 0V
Here instead we relax the pain and we create analog intermediate voltages, so now the output is an analog output

The output of a FLASH ADC should feed a standard digital encoder (where the components of the encoder are AND gate, OR gate and so on), here instead the outputs should feed an ANALOG ENCODER which checks if the voltage is above or below 0, so checks which comparators are triggered and which are not, also considering the intermediate analog voltages

⇒ The "ANALOG" ENCODER checks the zero-crossing !!!

ADVANTAGES:

1. 1/4 of comparators w/r t a standard flash ADC are used
   ↳ at least 1/4 less of power consumption

(2) Since we don't need to have a very high gain anymore, we don't need 25 transistors per comparator, let's say we need the gain to be very soft, so each comparator will have a lower # of TRANSISTORS

↳ much lower amount of power gets dissipated
→ also the INPUT parasitic capacitance will be much more reduced but the offset can be more relaxed

(3) HUGE AMOUNT OF AREA SAVING!!!

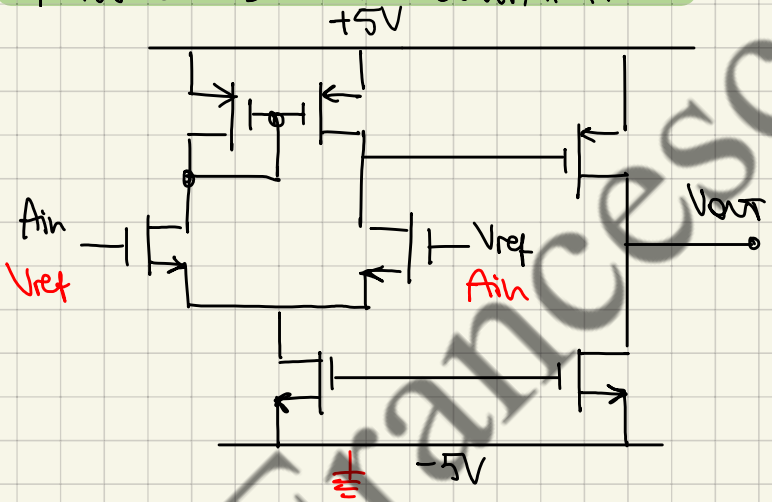
(4) Also dynamic performances are improved
In a flash ADC when we vary Ain we have to wait until the input of each of the 256 comparators settles to the new Ain value

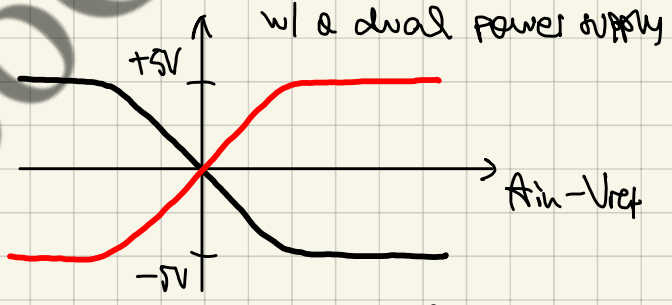
The interpolating ADC could work faster than a flash ADC in principle, but instead don't bc the gain gets relaxed, we use a much more complex ENCODER

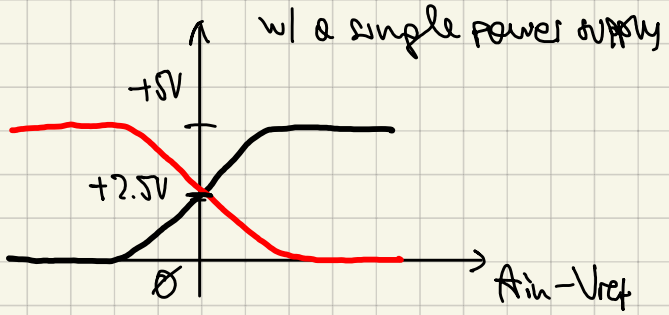
A further improvement could be using a DIFFERENTIAL COMPARATOR and two different ladders at the output.

Let's proceed step by step.
up to now we considered a comparator like this:

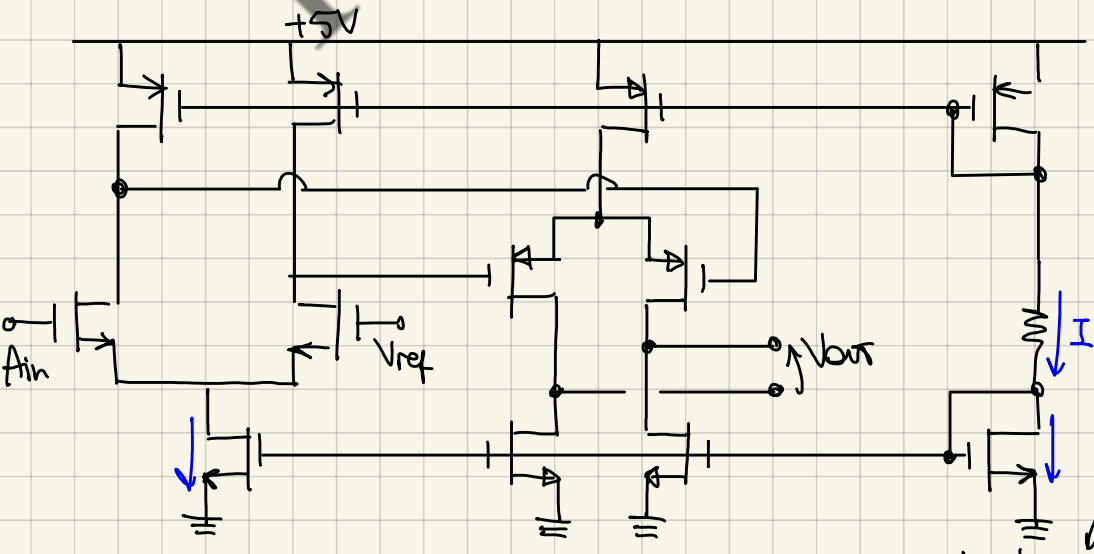
SINGLE ENDED OUTPUT COMPARATOR:

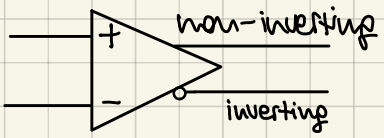


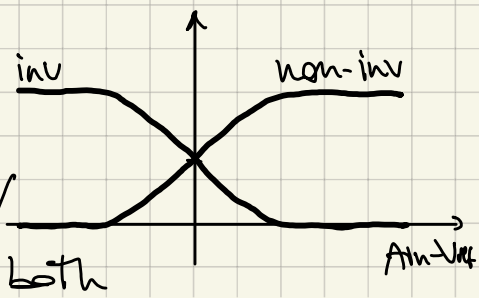




DIFFERENTIAL OUTPUT COMPARATOR



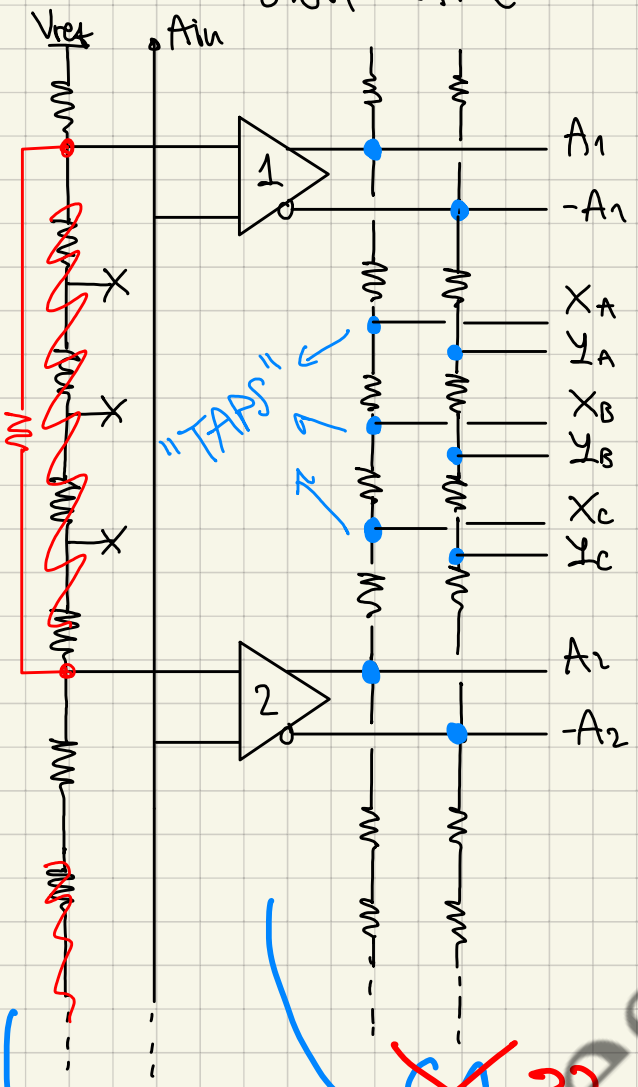




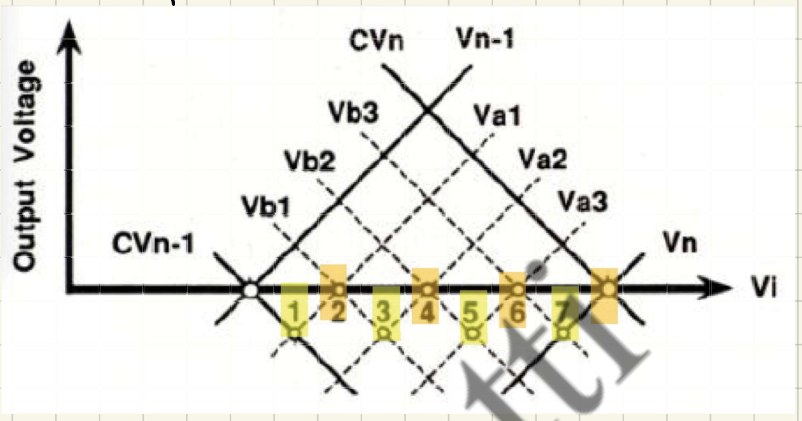
it has both

ADVANTAGE: Using differential comparators we can further halve the number of comparators and so also of the input resistors

3 bit - ADC



→ here we have 2 sets of 4 interpolation resistors



Interpolation resistors produce intermediate levels: 4 resistors → 2 intermediate levels

↓

w/ 4 resistors per side, we have:

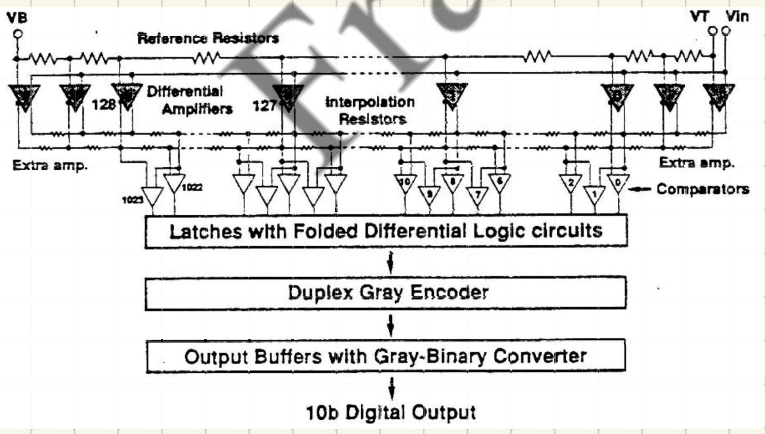
M=8 INTERPOLATION FACTOR

↳ corresponds to 3 EXTRA BITS

~~64~~ 32 DIFFERENTIAL "COMP.s"

~~64~~ 32 INPUT RESISTORS

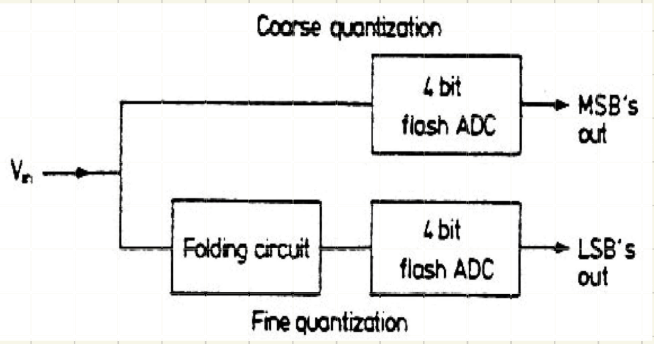
EXAMPLE: INTERPOLATING 10-ADC W/ DIFFERENTIAL COMPARATORS



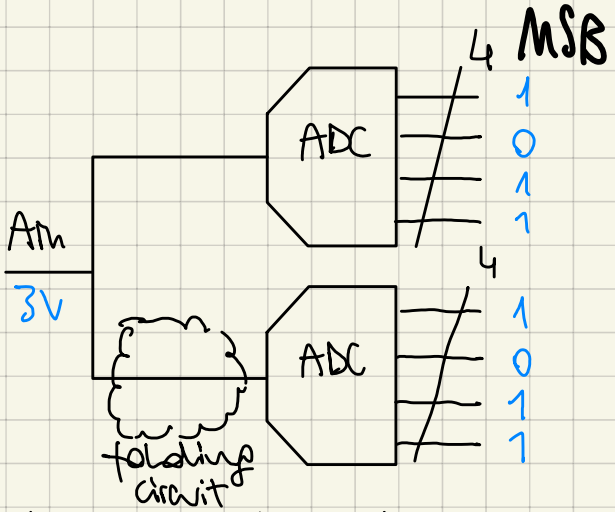
$2^{10} = 1024$ ⇒ using a standard flash ADC we should use 1024 input resistors and 1024 standard comparators

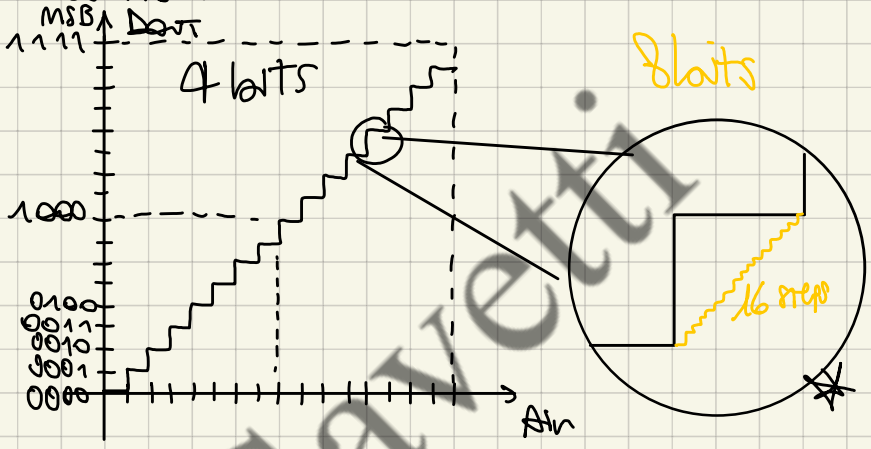
Instead, using an interpolating ADC w/ differential input comparators we'll use 1024/8 = 128 input resistors and 128 comparators

FOLDING ADC (FLASH FOLDING)



BASIC IDEA: by means of an analog signal pre-processing we can separately obtain the MSBs and the LSBs. The former are obtained w/ a low resolution ADC whereas the latter is obtained by sending the input signal into the folding circuit that translates the input ramp into a saw tooth, w/ limited maximum excursion

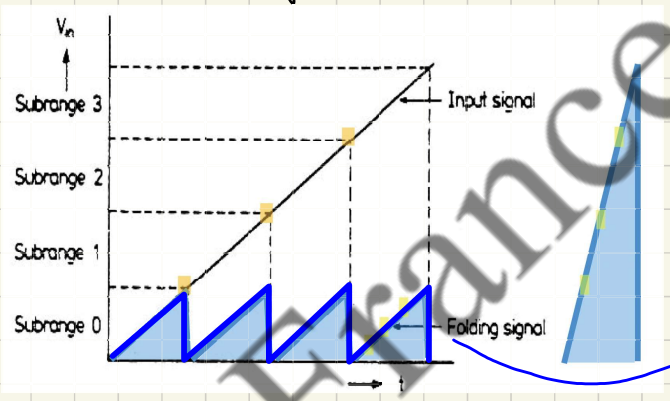




* if we want to subdivide each step by 16 levels we need something that converts Ain into the error btw the coarse quantization and the straight line

↓

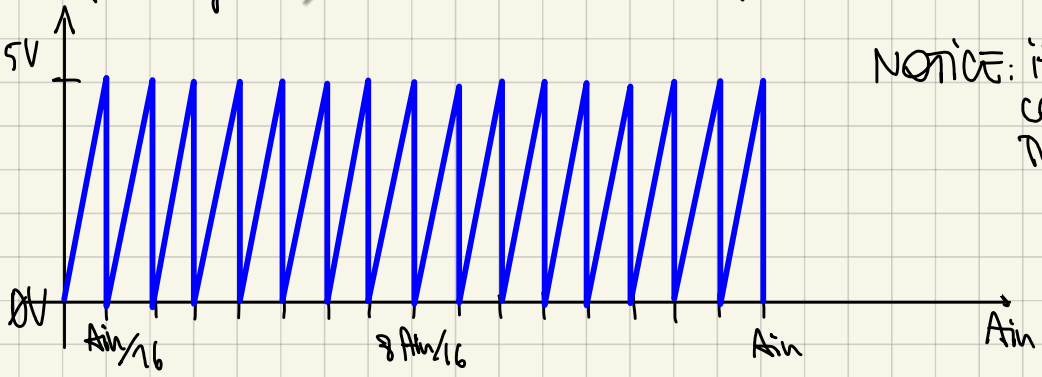
what we need is a FOLDING CIRCUIT which is a circuit whose input-output relationship is the blue one:



If the folding circuit is able to split the range btw 0 and 5V into 16 folds, then the height of the folding circuit will be the error btw Vin and the quantization achieved w/ the 16 levels ADC

SAW TOOTH UP-AND-DOWN CURVE

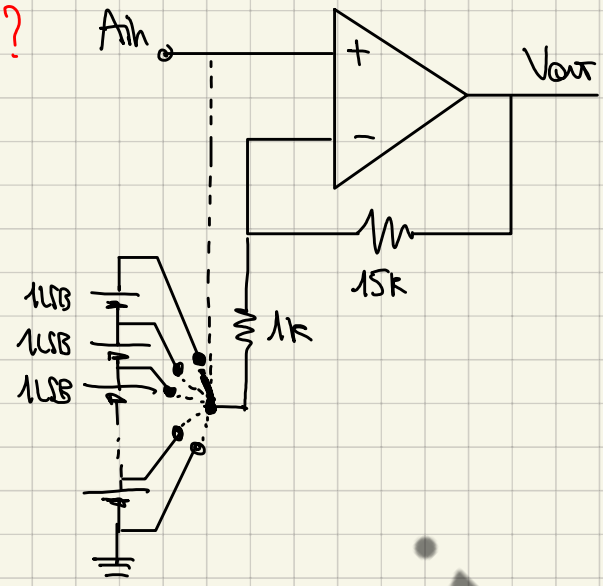
The folding circuit should be able to provide this in-out conversion



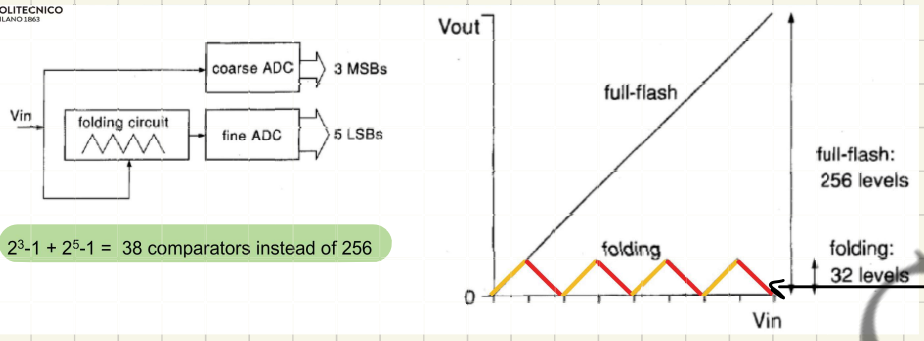
NOTICE: it gives us the error btw a coarse quantization and the actual Ain

Notice: This solution requires $2^4 - 1 + 2^4 - 1 = 15 + 15 = 30$ comparators

How could we design the folding circuit?
In principle, it's easy, it could be:

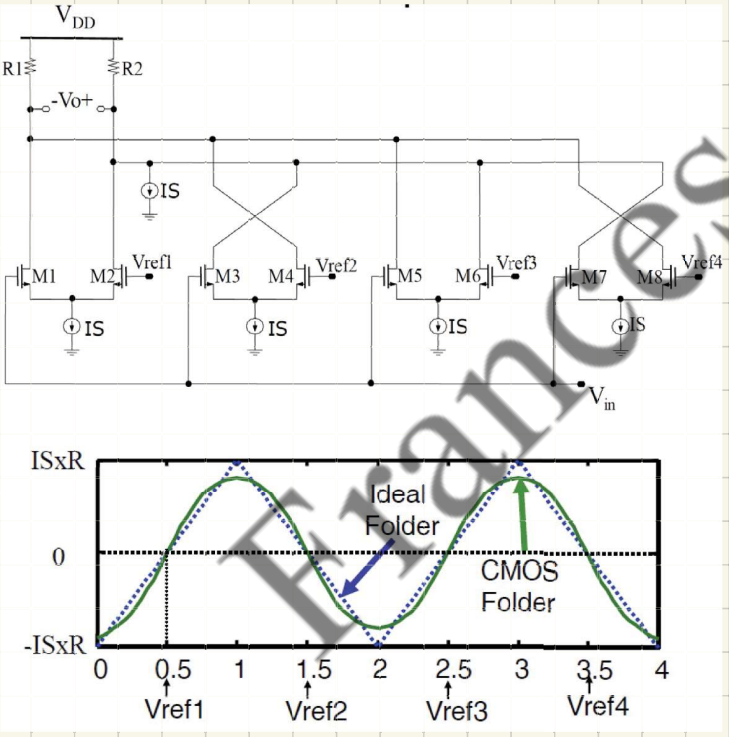


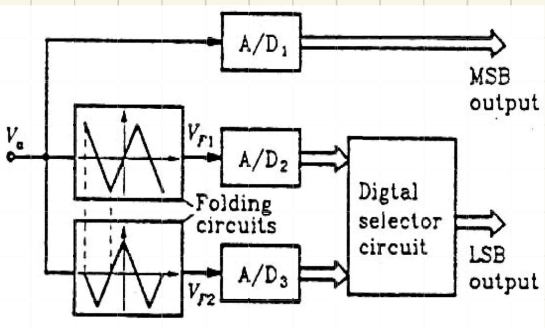
ANOTHER FOLDING FLASH ADC



The folding circuit instead of being a kind of saw tooth, is

POSSIBLE FOLDING FLASH ADC IMPLEMENTATION:





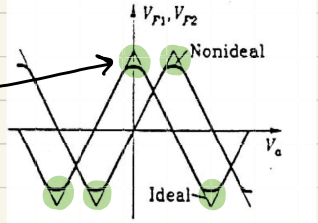
Instead of using just one folding circuit, we could use two of them: one shifted by the other by 1/2 LSB

↓

in this way we solve this issue of the smoothing in the curve

BUT we'll use the LSB coming from only one of the two

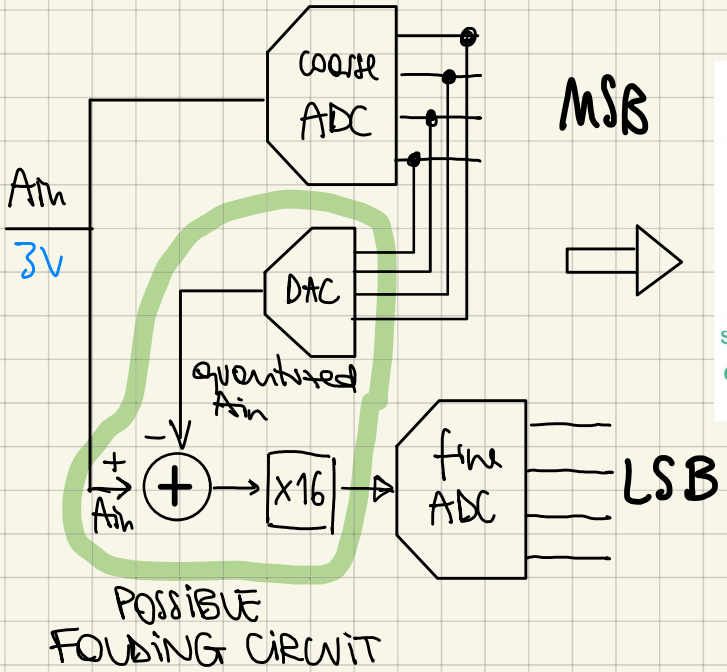
Which one? The one that gives the lower LSB

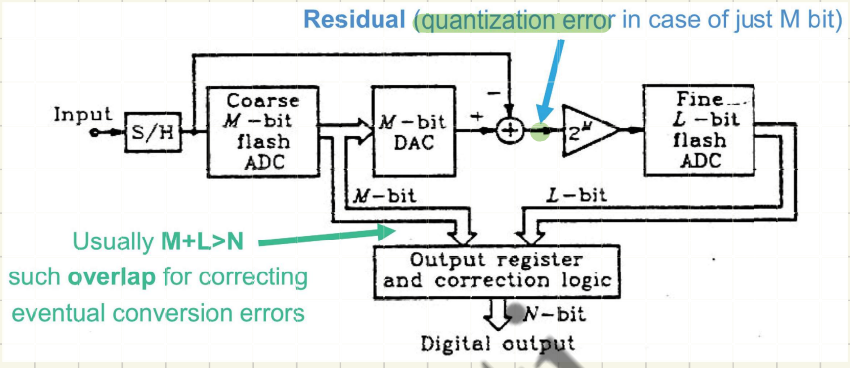


HALF-FLASH ADC

it comes from the folding ADC

We've seen that the folding ADC provides us the error of the coarse quantization





During the 1ST step the signal is applied to the 1ST flash converter which makes a low resolution conversion (coarse quantization) obtaining the M-MSBs.

These are converted by a DAC, whose accuracy must be at least equal to that one of the whole ADC, ie N bits.

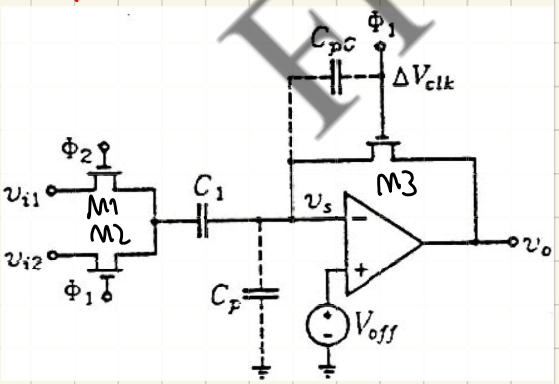
This value, which is an approx. of the input signal, is subtracted from the same input signal and the result is the RESIDUAL, ie the error committed approximating the signal w/ M bits.

The residual is amplified and sent to the 2nd flash ADC that determines the L-LSBs doing the FINE QUANTITATION.

The outputs of the converters are summed and the result is the N bits digital output.

We desire to have M+L>N in order to perform the correction of any conversion error → the additional bits are named OVERLAP BITS

INTERNAL DESIGNS OF HALF-FLASH ADC



COMMUTATING AUTO-ZEROING TECHNIQUE

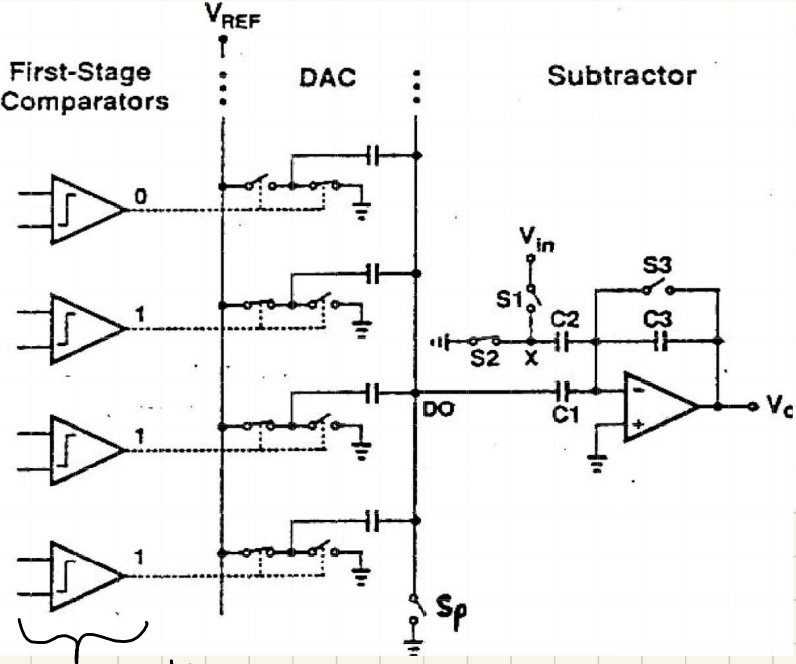
1st phase: { M1, M2 open → the comparator acts as an OpAmp connected as a buffer
M3 close

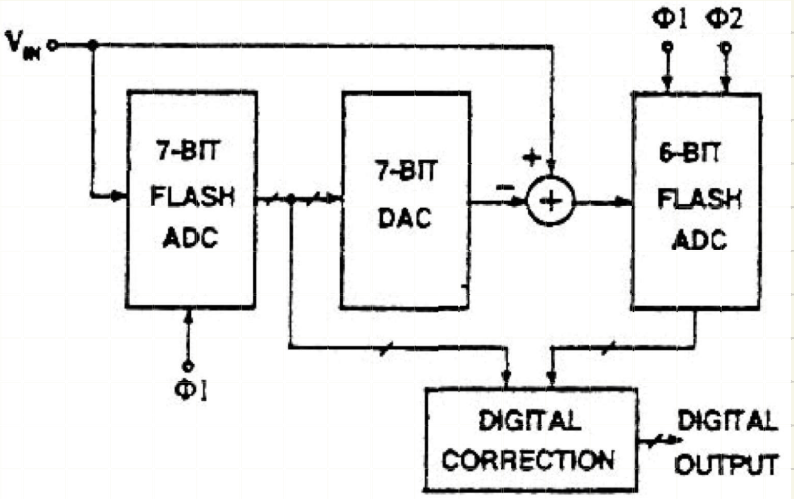
↳ Then thanks to the FB, V_{OS} is applied to C_1

2nd phase:
so closing M2 and applying $V_{i2} = 0$, we store V_{OS} across C_1 → AUTO-ZEROING

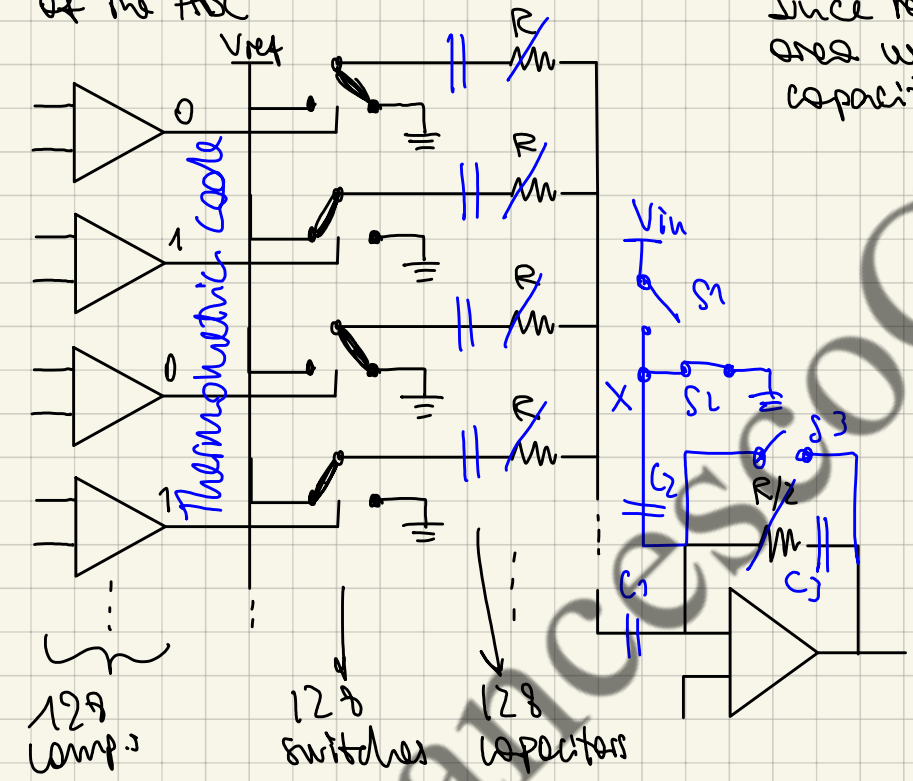
3rd phase { M1 close
M2, M3 open
and we apply $V_{i1} = V_{ref}$ which will be compared to ground

The comparator suffers of its own V_{OS}, but we stored it across C_1 so the two compensate each other





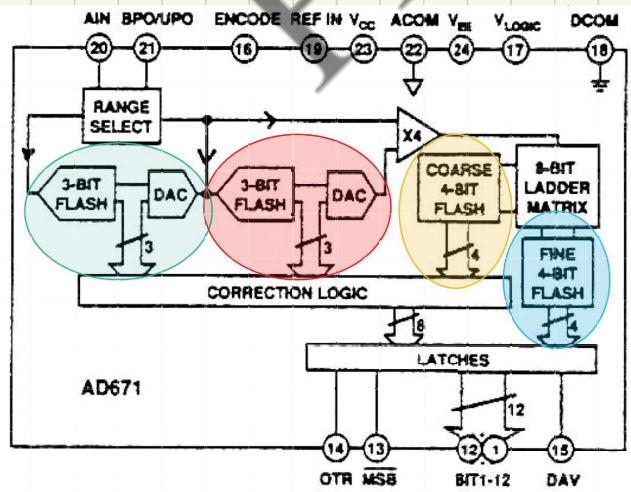
comparators of the ADC

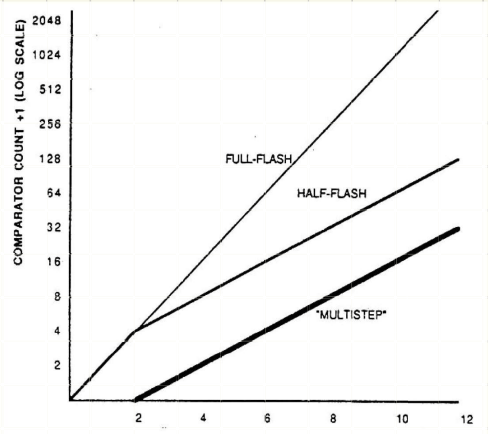


Since resistors require a huge amount of area we can substitute the resistors w/ capacitors

MULTISTEP FLASH ADC → we have a cascade of more than 2 flash ADCs

12 - ADC

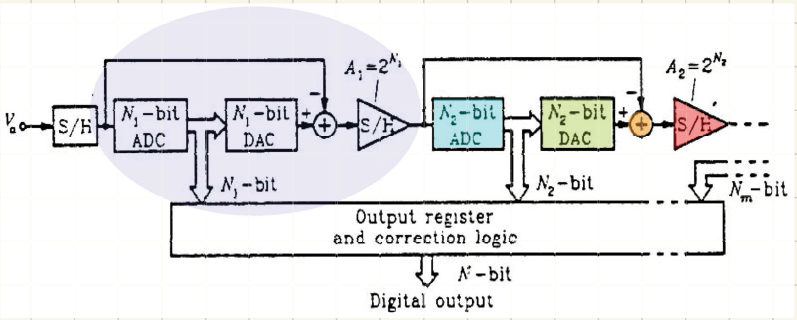




PIPELINED ADC

We've seen that the half-flash and the multi-step conformations allow, in respect of flash ADCs, to greatly minimize the area occupation, the power consumption and the conversion speed.
At the same time, many applications require high resolution and high sampling frequency.

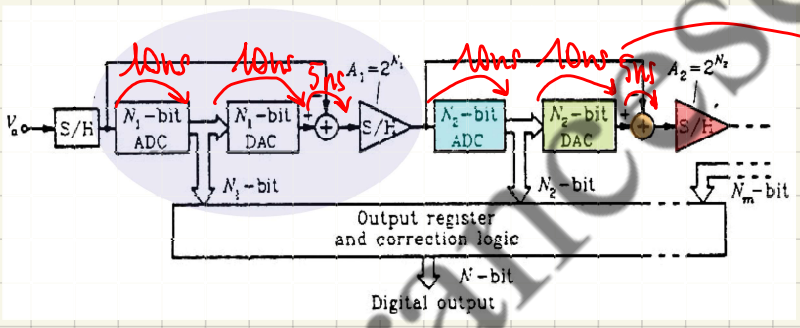
⇒ The demand for high sampling freq, low area occupation and moderate power consumption is fully satisfied by using PIPELINED CONVERTERS.



A typical PIPELINED ADC is composed of m stages each of which contains:
- an S&H amplifier
- a low resolution flash ADC (1÷4 bit)
- a DAC
- an analog adder

From one stage to the other there always be a S&H, in this way after the conversion of the i-th stage, the residual which reaches the (i+1)-th stage is maintained by the S&H.
In doing so, the i-th stage can reach a new sample

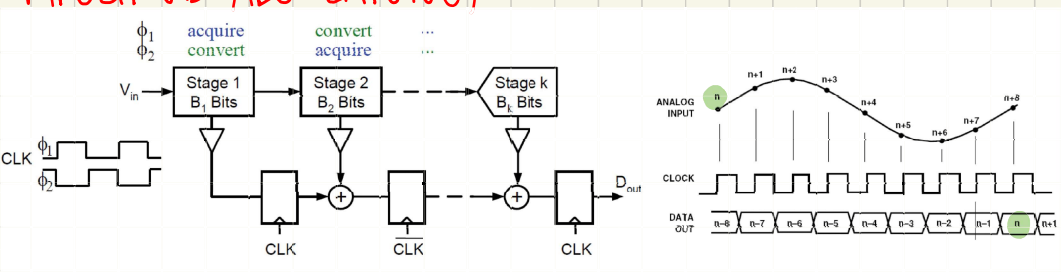
NOTICE: A pipilined architecture is a multistep ADC where each stage has sufficient time to convert, bc it's not converting something that is propagating asynchronously through the network, but each stage is converting something that is kept constant by the previous S&H



in a 4 blocks ADC, one sample takes 100ns to convert (in a flash ADC it took just 10ns)
BUT, at every clock pulse we can obtain my output bc we can put together the result of the 1ST coarse conversion w/ that one of the coarse conversion that happens 25ns later and so on

⇒ In a pipelined ADC w/ m stages the sampling frequency is equal to the clock frequency, while every conversion requires m clock cycles, and therefore the code for a certain sample appears at the output after a certain delay named LATENCY or PIPELINE DELAY

PIPELINED ADC LATENCY



The output conversion of the n-th sample will be available after a given amount of clock cycles equal to the # of stages in which the ADC is divided

⇒ We need to properly choose the # of stages

ADVANTAGE: we've got an output every Tclock

ISSUE: LATENCY: using 8 stages, the output related to the i-th sample will be available after 8 Tclock (= Tconv)

↓

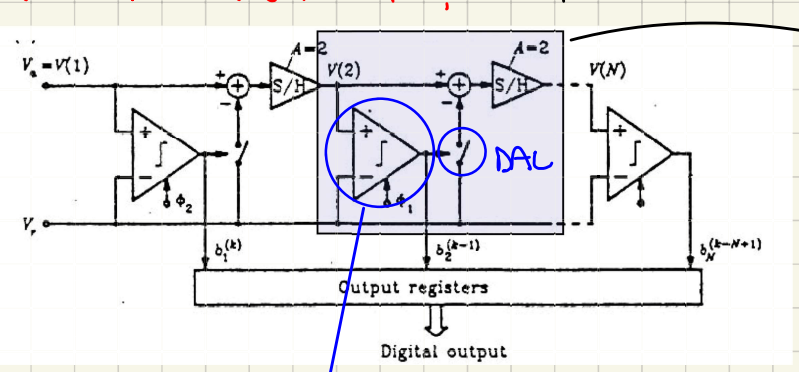
but it's not a real issue bc7 we'll make the ADC operating in FREE-RUNNING mode

"EXTREME" IMPLEMENTATION OF A PIPELINED ADC

we wanna design a 12 bit ADC, instead of dividing them in 4+4+4, we use 12 stages of 1 bit each

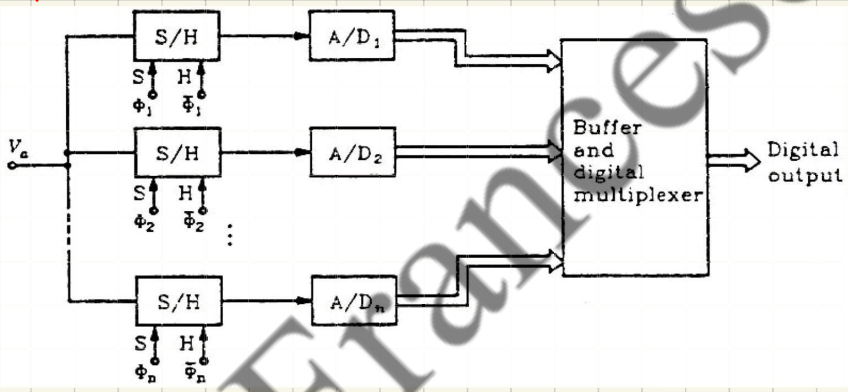
What's a 1 bit ADC? → it's a comparator

What's a 1 bit DAC? → it's a switch

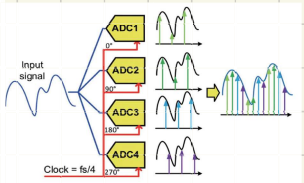


1 bit ADC = comparator

TIME-INTERLEAVED ADC



To obtain an high sampling freq. we can use two or more ADCs connected in parallel, obtaining a structure named the "TIME INTERLEAVED ARRAY"



This structure is made of C channels in each of which is present a S&H and a converter w/ N bits

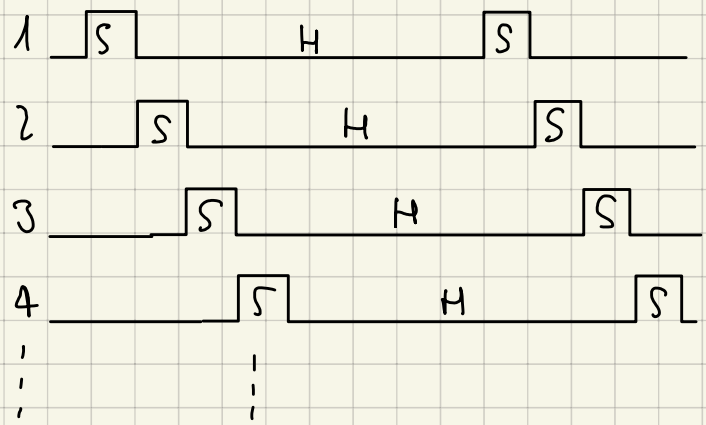
In every channel the input signal is sampled w/ $f_s = 1/C T_{clock}$

The channels work sequentially, therefore the sampling frequency of the whole ADC is n times greater of that one of the single converter whereas the resolution and the maximum frequency of the input signal are unchanged

example: if we want f_s = 1Gsps w/ a 10-ADC it means that in principle T_{conv} must be $T_{conv} = 1/f_s = 1ns$ which is impossible to obtain w/ a 10 bit ADC

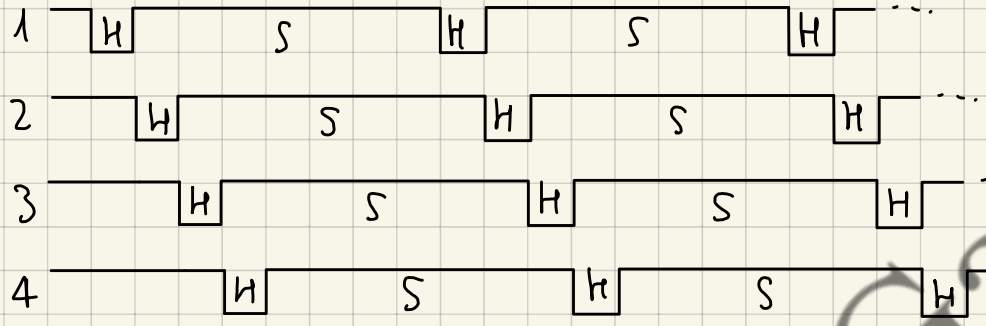
→ we use 10 channels, each of them composed by a S&H and a 10-ADC w/ T_{conv} = 10 ns

① POSSIBLE IMPLEMENTATION



→ This is a very stupid approach bcz during sample the switch is closed for a very short time and so the S&H should be very very fast in acquiring the signal

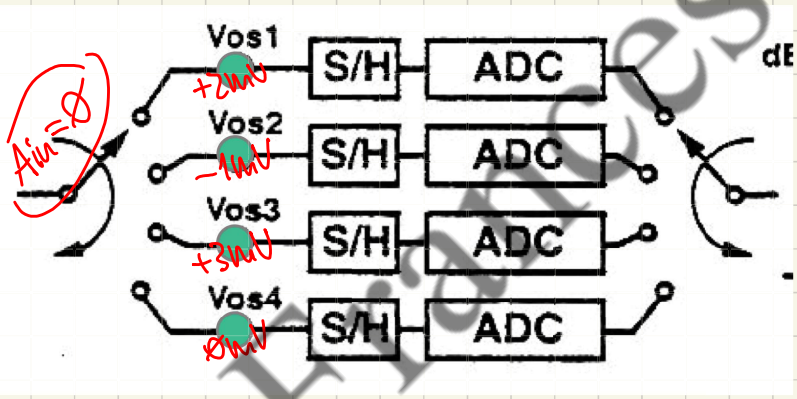
② POSSIBLE APPROACH → BETTER!!



$T_H \geq T_{conv,each\,ADC} = 10ns$

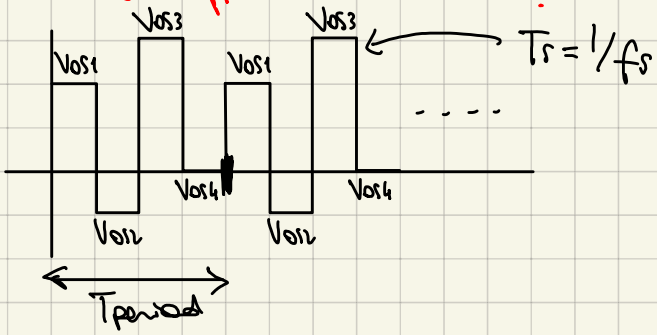
ERRORS:

① OFFSETS



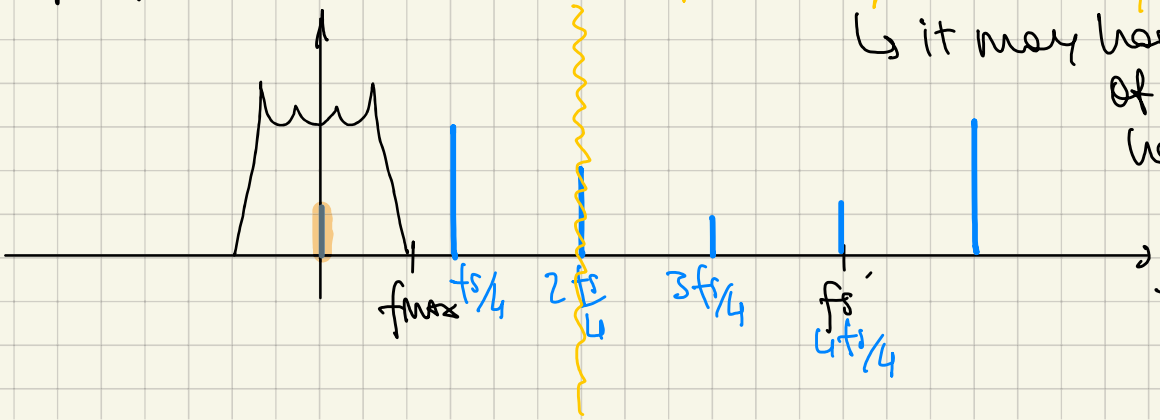
TYPICAL EXAM'S QUESTION

What's the error that would you read out from the 4 ADCs?



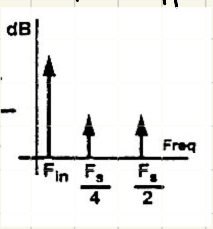
output signal: $f = \frac{1}{T_{period}} = \frac{1}{4T_s} = \frac{1}{4} f_s$

output spectrum



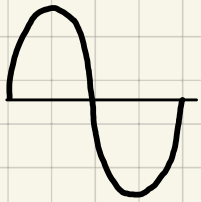
↳ it may happen that one of the following harmonic falls inside our baseband

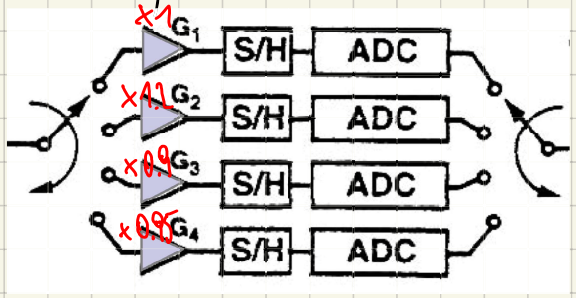
⇒ An offset error in the ADC appears as a tone at f_s/C, $2f_s/C$, $3f_s/C$, ...



② GAINS

Imagine that every S&H has its own gain

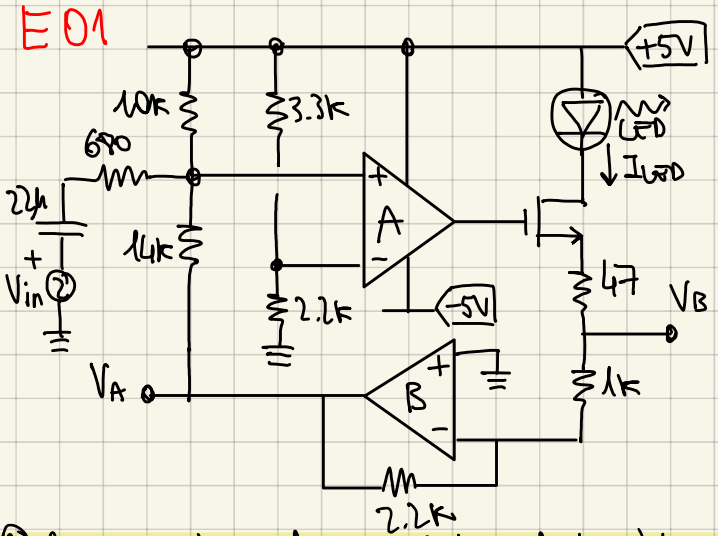




TO FINISH!!!!

ES15 - EXERCISES
01/12/2021

E01



OpAmps: $A_0 = 50V/mV$

$V_{OS} = 5mV$

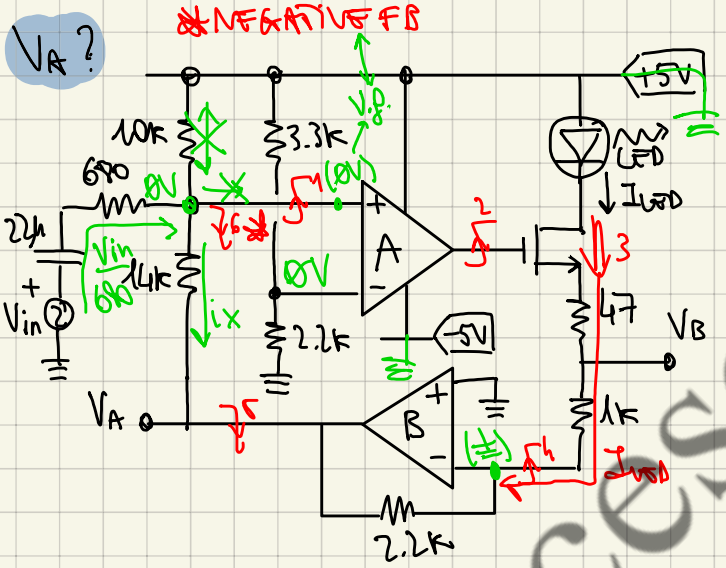
$I_B = 200pA$

MOSFET: $k = \frac{1}{2}\mu_n C_{ox} = 10mA/V^2$

$V_T = 0.8V$

Ⓐ Compute the relationships of V_A, V_B and I_{LED} vs. Vin

SMALL SIGNAL ANALYSIS



The OpAmp A has an amplification of A_0 while the OpAmp B has an amplification of $\left(-\frac{2.2K}{1K}\right)$ bc it's an inverting config.

we proceed in anti-clockwise direction

signal

When we study Vin we have to turn off all the other sources, included the PS. To perform the small signal analysis

@DC → C open → $\frac{V_A}{V_{in}}(0) = 0$

@HF → C short → $V_A = -V_{in}\frac{14K}{690} \Rightarrow \frac{V_A}{V_{in}}(\infty) = -20.6\ \frac{V}{V}$

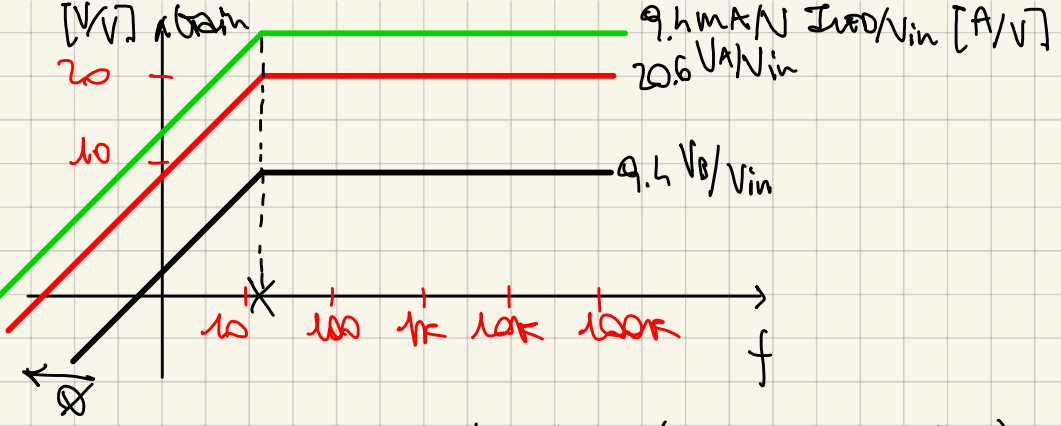
$V_B = ?$

@DC → $\frac{V_B}{V_{in}}(0) = 0$

@HF → $V_A = V_B\left(-\frac{2.2K}{1K}\right) \rightarrow -20.6V_{in} = V_B(-2.2) \Rightarrow \frac{V_B}{V_{in}}(\infty) \simeq 9.4\ \frac{V}{V}$

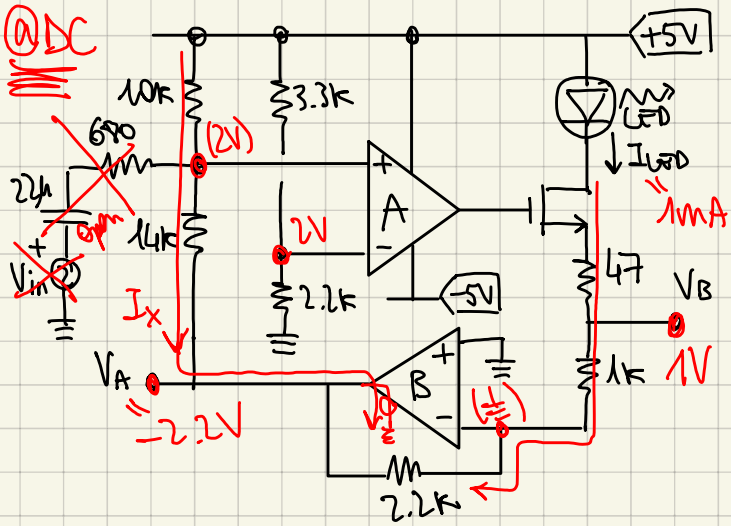
$I_{LED} = ?$

$i_{LED} = \frac{V_B}{1K} \Rightarrow \frac{i_{LED}}{V_{in}}(\infty) = \frac{9.4}{1K} = 9.4\ mA/V$



LET'S STUDY THE POLARIZATION (THE BIAS POINT) (when $V_{in}=0$)

@DC



$V_A = 5 \frac{2.2K}{5.5K} = 2V$

$I_x = \frac{5-2}{10K} = \frac{3}{10K} = 0.3mA = 300\mu A$

$V_A = 5V - 300mA \cdot 2K = -2.2V$

$V_A = V_0 (-2.2) \Rightarrow V_B = \frac{V_A}{-2.2} = \frac{-2.2}{-2.2} = 1V$

$I_{LED} = \frac{V_B}{1K} = \frac{1V}{1K\Omega} = 1mA$

$V_S = V_B + i_{LED} \cdot 47 = 1V + 47mV = 1.047V$

$i_d = k(V_{GS} - V_T)^2 \rightarrow V_G = V_S + V_T + \sqrt{\frac{i_d}{k}} = 1.847V + \sqrt{\frac{1mA}{100mA/V^2}} = 2.16V$ (0.316)

POLARIZATION VALUES

$V_A = -2.2V$
$V_B = 1V$
$I_{LED} = 1mA$

SATURATION CHECK:

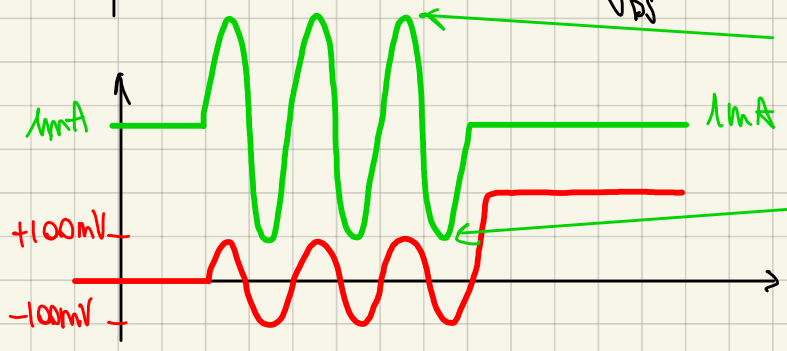
saturation region →

$V_{GS} \geq V_T$ ✓

$V_{DS} > V_{OV}$ → $V_D = 5V - 0.6 = 4.4V$, $V_S = 1.847V$

→ $4.4V - \cancel{1.847V} > 2.16 - \cancel{1.847} - 0.3$ ✓

YES! The MOS effectively works in saturation



$1mA + 9.4\frac{mA}{V} \cdot 100mV = 1.94mA$

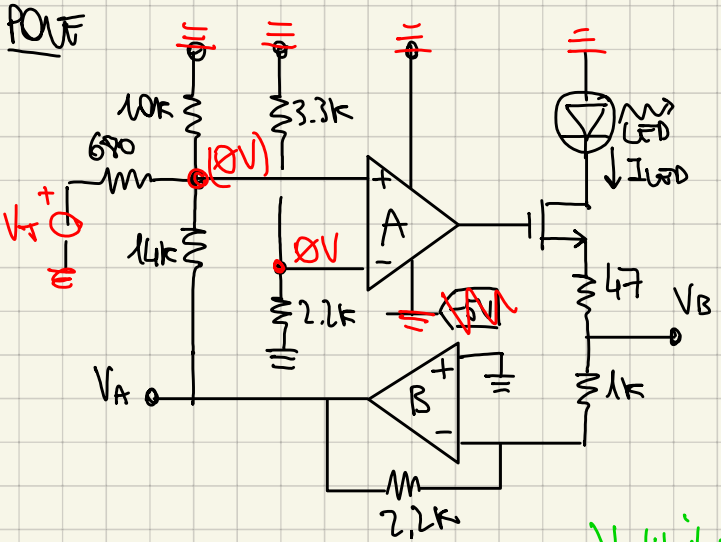
$1mA - 9.4\frac{mA}{V} \cdot 100mV = 0.06mA$

$$i_{D,MAX} = K\left(V_{GS,MAX} - V_T\right)^2 = 10\,\frac{mA}{V^2}\left(5V - 0.8V\right)^2 = 176\,mA$$

$$i_{D,min} = 0A \quad (\text{bc cannot invert})$$

Ⓑ Compute the input pole and plot $V_A(t)$ and $I_{LED}(t)$ waveforms when the input is a high frequency 200mV$_{pp}$ sinusoid

POLE



$$f_p = \frac{1}{2\pi \cdot 22\mu F \cdot 680\Omega} = 10.6\,Hz$$

Beware: small signal → $V_A(t)$

$$\frac{V_A}{V_{in}}(\infty) = 20.6$$

100mV · 20.6 = 2.06V

DC contrib (no variation in time)

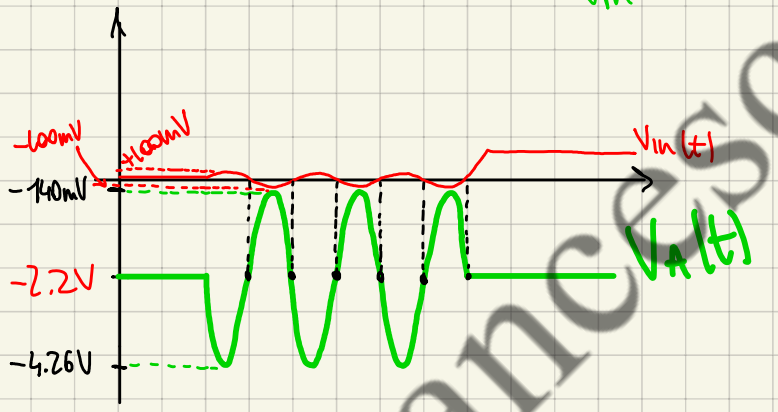
+100mV

-100mV

$V_{in}(t)$

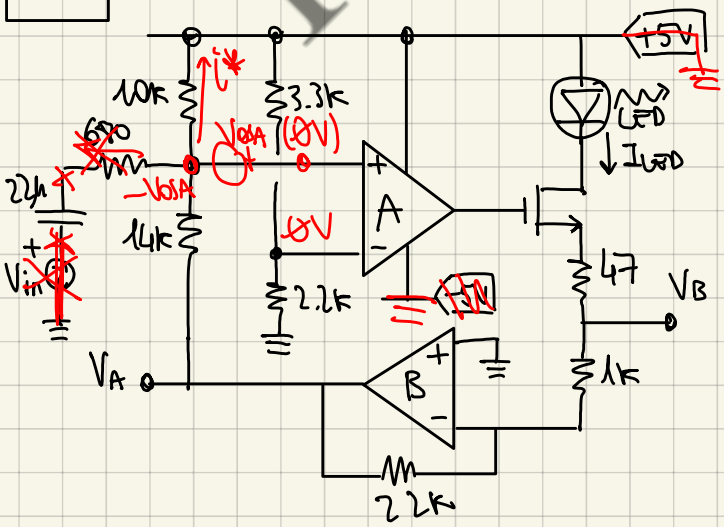
$V_A(t)$ it stays to zero bc

$\frac{V_A}{V_{in}}(0) = 0$



Ⓒ Compute the maximum V_A static error due to V_{OS} and I_B of the OpAmp

V_{OSA}:



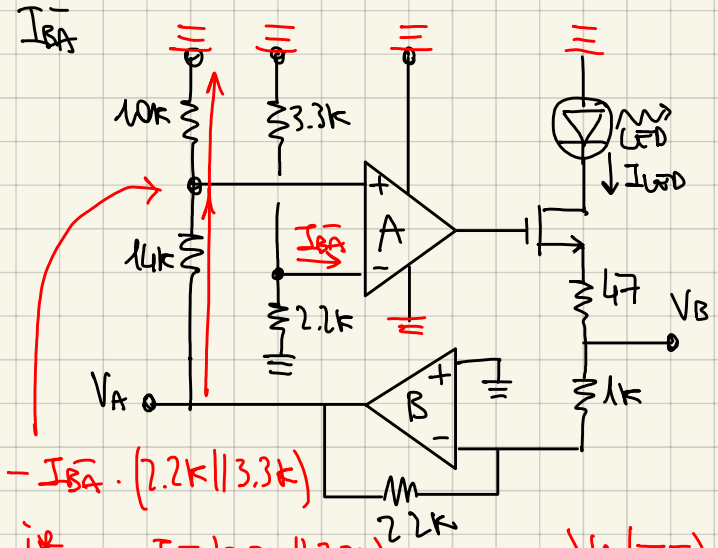
$$i^* = \frac{V_{OSA}}{10K}$$

$$V_{A,OS} = \frac{V_{OSA}}{10K} \cdot 24K = \pm 5mA \cdot 2.4 =$$

$$= \pm 12mV$$

$$V_{B,OS} = \frac{\pm 12mV}{-2.2} = \mp 5.5\,mV$$

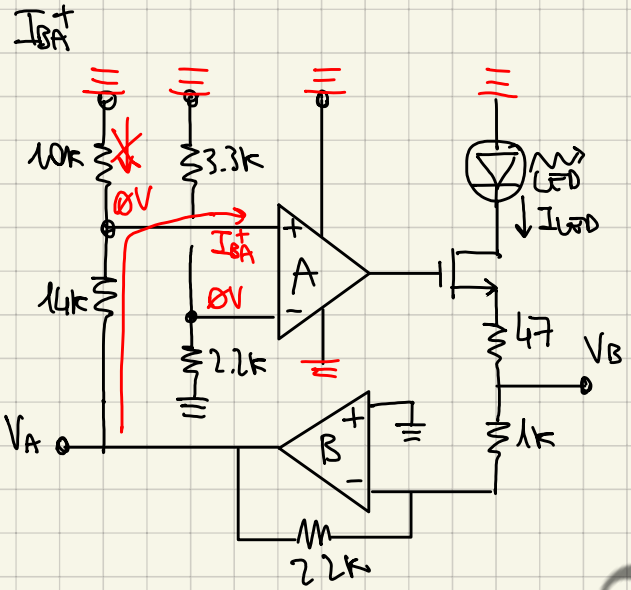
$$I_{LED,OS} = \frac{\mp 5.5mV}{1K} = \mp 5.5\,\mu V$$



$-I_{BA}^- \cdot (2.2K \| 3.3K)$

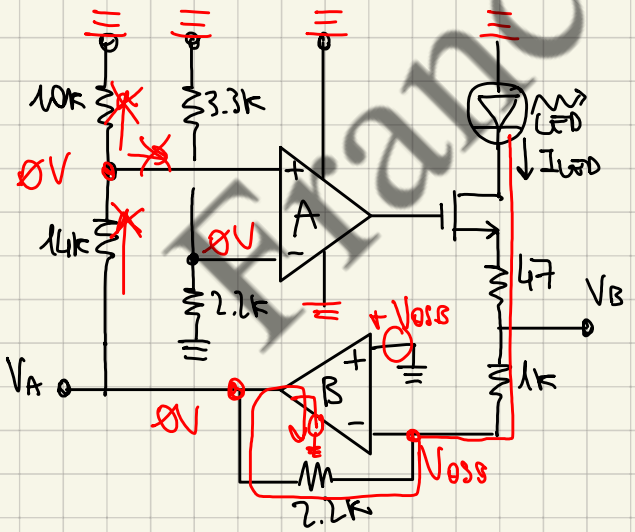
$i^* = \dfrac{-I_{BA}^- (2.2K \| 3.3K)}{10K} \rightarrow V_A(I_{BA}^-) = \dfrac{-I_{BA}^- (2.2K \| 3.3K)}{10K} \cdot 2K \approx$

$\approx -0.63\,\mu V$ negligible w.r.t. $\pm 12mV$

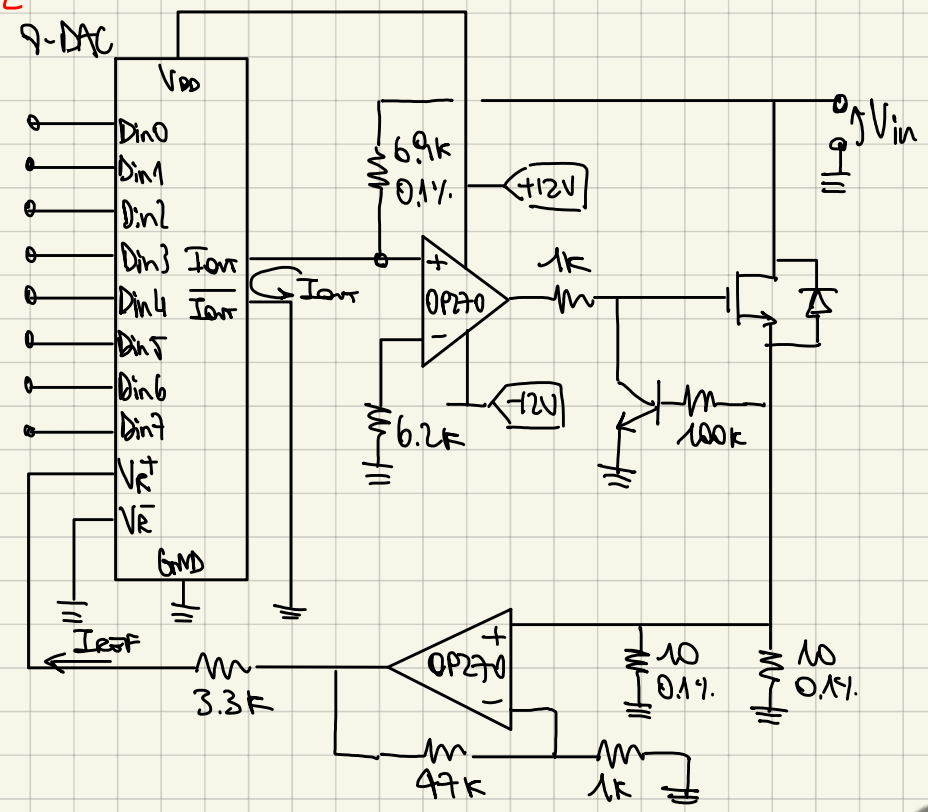


$V_A(I_{BA}^+) = I_{BA}^+ \cdot 14K = 200pA \cdot 14K\Omega = 2.8\,\mu V$

negligible



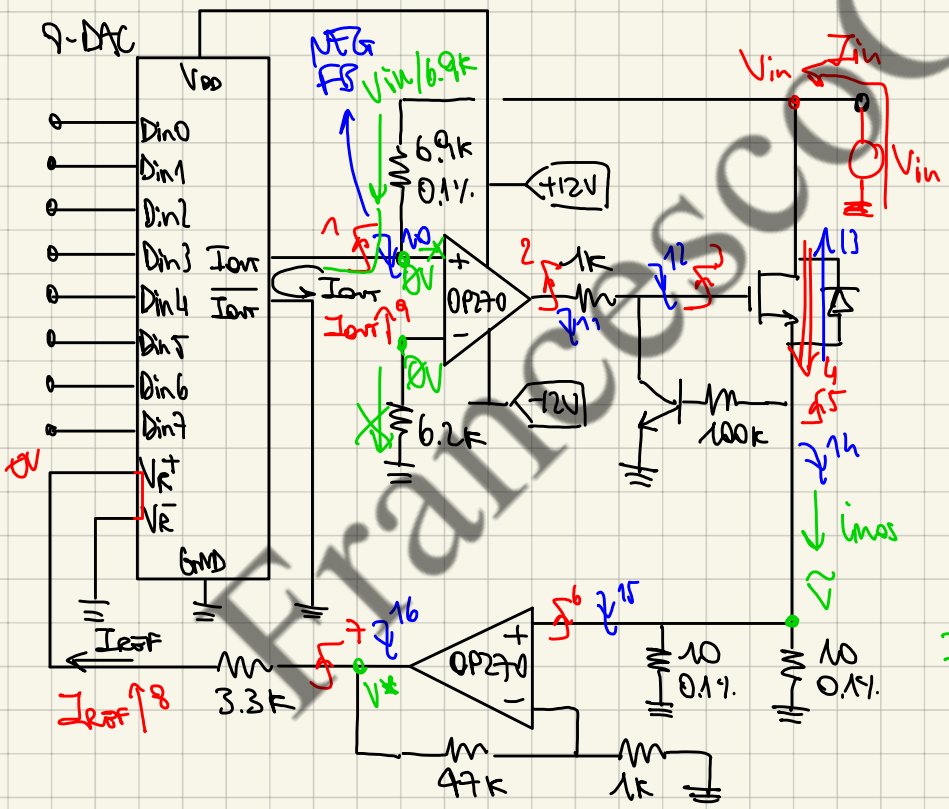
E02



DAC provides:
$I_{out} = I_{ref} \cdot Din/256$
and V_{ref} @ V_R^+ input

Opamps: $A_0 = 150dV/mV = 1.5M$
$GBWP = 5MHz$

Ⓐ Obtain the analytical relationship V_{in}/I_{in}, as a function of N



$$I_{out} = \frac{V_{in}}{6.9K} = I_{ref}\frac{Din}{2^8}$$

$$\Downarrow$$

$$I_{ref} = \frac{V_{in}}{6.9K}\frac{2^8}{Din} = \frac{V_{in}}{Din}\cdot\frac{1}{27.2}$$

$$V^* = I_{ref}\cdot 3.3K$$

$$V^* = \tilde{V}\left(1+\frac{47K}{1K}\right)$$

$$\tilde{V} = \frac{V^*}{49} = \frac{V_{in}}{Din}\frac{3.3K}{27\cdot 49} = \frac{V_{in}}{Din}2.5$$

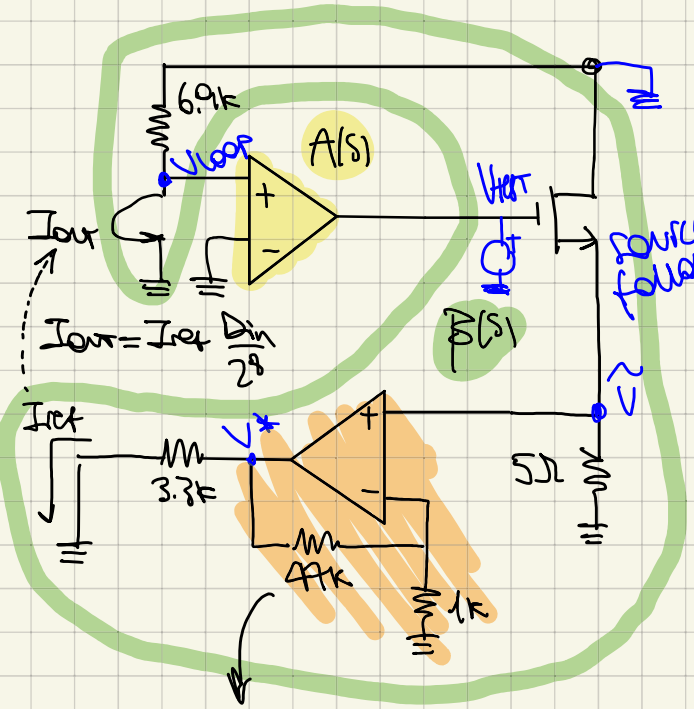
$$I_{mos} = \frac{\tilde{V}}{10||10} = \frac{\tilde{V}}{5} = \frac{V_{in}}{Din}\cdot\frac{1}{2}$$

$$I_{in} = I_{mos} + \frac{V_{IN}}{6.9K} = \frac{V_{IN}}{DIN}\frac{1}{2} + \cancel{\frac{V_{IN}}{6.9K}}$$ (negligible)

$$\Rightarrow \frac{V_{IN}}{I_{IN}} = 2.5\cdot Din = Req$$

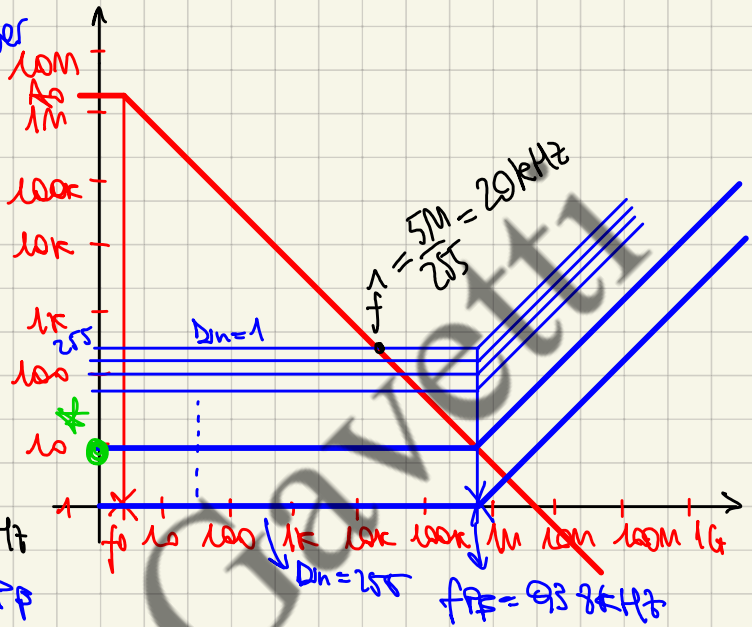
Din = 0 → Req = 0Ω
Din = 1 → Req = 2.5Ω
⋮
Din = 128 → Req = 256Ω
⋮
Din = 255 → Req = 510Ω

(B) Reckon if the stage is stable or not when Din = 255. Moreover, tell if mobility improves by reducing Din (hint: assume $1/g_{mM91} = 495\,\Omega$ and ignore the role of the BJT)



$A_0 = 1.5M = 20\log_{10}(1.5\cdot10^6) = 123\,dB$

$f_0 = \frac{GBWP}{A_0} = \frac{5M}{1.5M} = 3.3\,Hz$



$GBWP = 5M,\ G_1 = 48,\ f^* = \frac{5M}{48} = 938kHz$ → f_{PP}

$\beta = \frac{5\Omega}{5\Omega + \frac{1}{g_m}} \cdot \left(1 + \frac{47k}{1k}\right) \cdot \frac{1}{3.3k} \cdot \frac{Din}{28} \cdot (-6.9k) = -\frac{Din}{255}$

$\frac{5\Omega}{5\Omega + \frac{1}{g_m}}$: $g_m = \frac{1}{0.01} = 12$; $\left(1 + \frac{47k}{1k}\right) = 48$

$\rightarrow \frac{1}{\beta} = -\frac{255}{Din}$ ⟨ ∞ if Din = 0 ; -255 if Din = 1 ; -1 if Din = 255

$\frac{1}{\beta_x} \cdot 938kHz = 5MHz$

$\frac{1}{\beta_x} = \frac{5M}{938k} = 48 \leq Din$ stable

if $Din < 48 \Rightarrow$ UNSTABLE

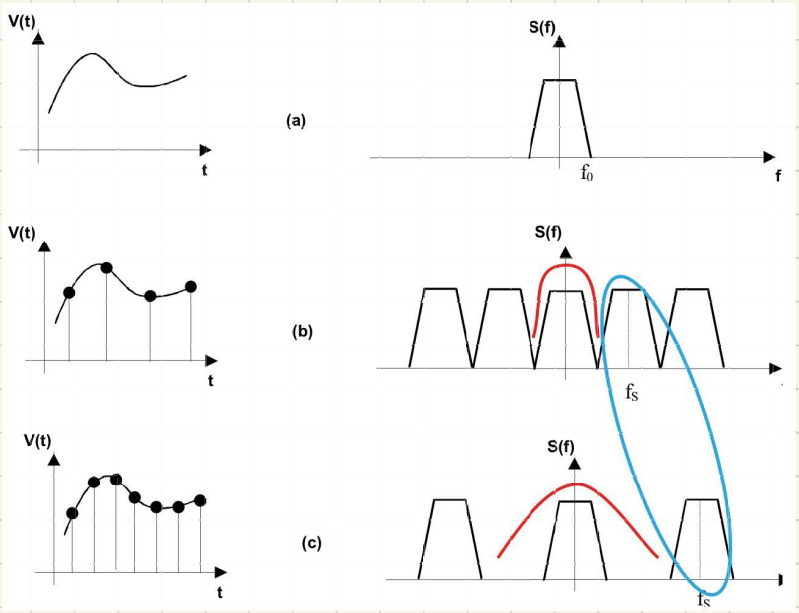
if $Din = 48$ (20–40 closure angle) ↓ $PM = 45°$

- why increasing fs beyond the Shannon minimum
- standard oversampling
- advanced sigma-delta oversampling
- why 1bit is equal or even better than many?
- need for DSP for digital filtering and decimation

ES15 - OVERSAMPLING AND SIGMA-DELTA

14/12/2021

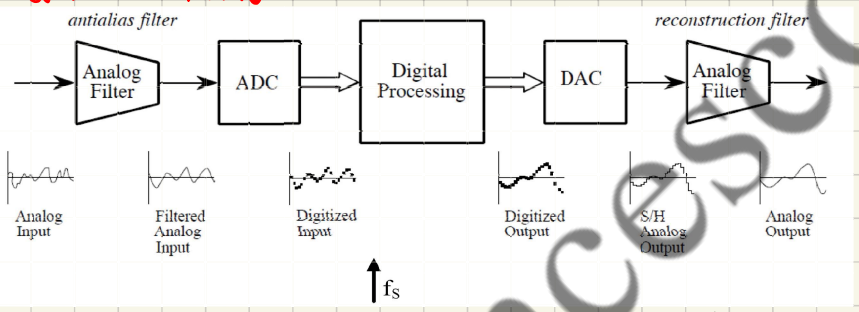
SAMPLING



if Ts is long → fs is low and so the replicas will be very close to each other or they can even overimpose if $f_s < 2 f_{signal}$
furthermore if we have to pass only the baseband and kill all the other replicas our LPF must be very selective

↳ decreasing Ts, we increase the sampling frequency fs, so the replicas will be more apart from each other and our LPF can be more relaxed (lower # of poles)

QUANTIZATION



The analog filter at the input and the one at the output must be identical bc they should let the signal pass up to fs killing all the other replicas

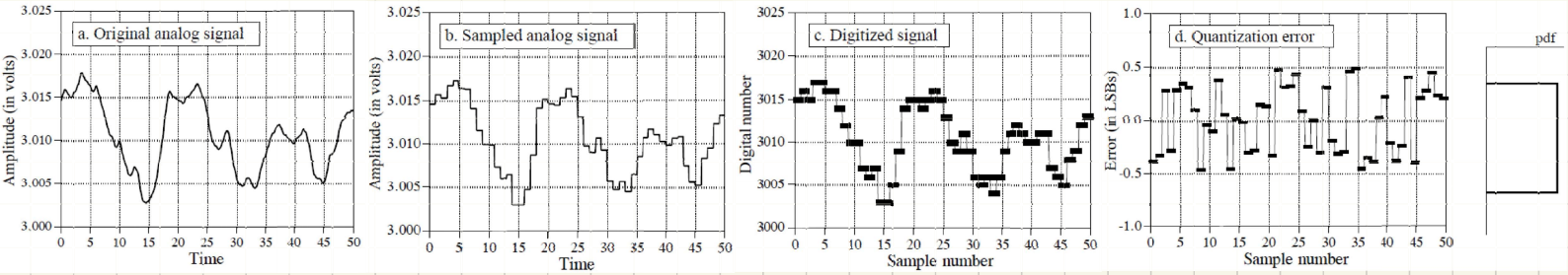
Due to the fact that the ADC provides a quantization of the vertical amplitude w/ steps equal to 1LSB, the ADC introduces an error into the signal, bc the signal that feed the DSP is not equal to the original one but it will be affected by the so called QUANTIZATION ERROR

$$\overline{e_q^2} = \left(\frac{FSR}{2^n\sqrt{12}}\right)^2 = \frac{LSB^2}{12}$$ POWER OF QUANTIZATION ERROR

$$Q_q(f) = \frac{\overline{e_q^2}}{f_s/2}$$ QUANTIZATION ERROR'S SPECTRAL DENSITY

↳ The spectrum is symmetric respect to fs/2

example: FSR=5V fs=40kSps $\overline{e_q^2} = (352\mu V)^2$ $Q_q(f) = (3.52\mu V/\sqrt{Hz})^2$

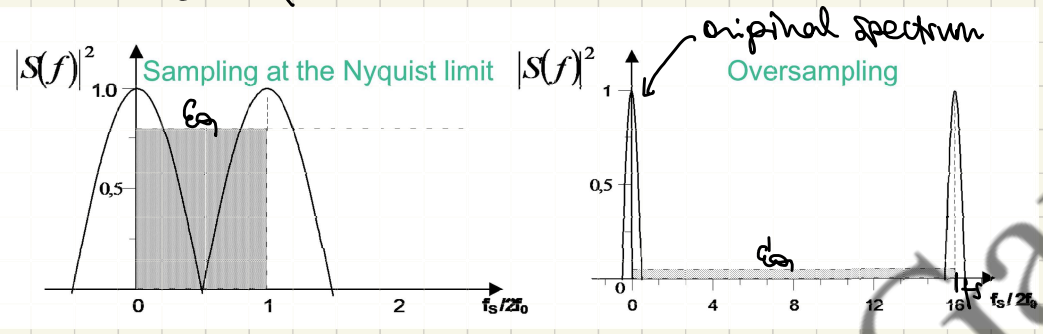


quantization error = sampled signal - digitized signal : $d = b - c$

if the ADC is well designed $|E_q| \leq \frac{1}{2}$ LSB

OVERSAMPLING

If we perform our sampling @ the Shannon/Nyquist limit $f_s = 2f_{max}$ we end up that the replicas touch @ f_{max} and E_q is distributed uniformly btw 0 and f_s

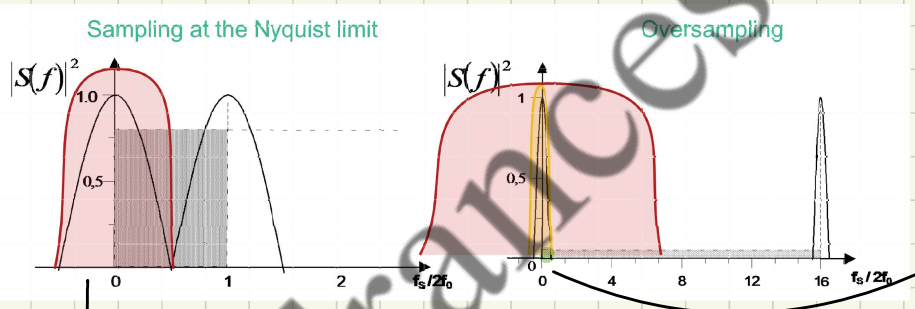


OVERSAMPLING FACTOR

$$OS = \frac{2f_0}{f_s}$$

Otherwise, if we employ $f_s \gg 2f_{max}$ (e.g. $f_s = 16/(2f_{max})$) the two replicas are very separated one another and E_q gets spread over a wider frequency range

$E'_q = \frac{E_q}{16}$ → we can get an improvement if we perform a smart filtering action



The small portion of E_q we collect using a smart filtering action

if $f_s = 2f_0 \Rightarrow BW_{LPF} = f_s/2$ (= f_0 in this case) → in this way we are able to kill all the replicas but we acquire all the E_q

OVERSAMPLING: even in this case we can use a LPF w/ BW = $f_s/2$ but again we reconstruct our original spectrum, killing all the replicas, but we also recollect all the E_q the ADC had introduced

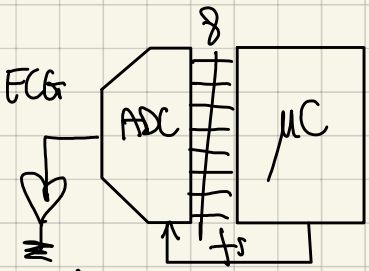
SMART FILTERING: let's filtering up to f_0 and not $f_s/2$.
In this way we are able to fully reconstruct our original signal, but we'll recollect only a small portion of E_q

we improve the SNR !!!

CLASSICAL ADC: $SNR_{Th} = \frac{V_{eff}}{q} = \frac{FSR/2\sqrt{2}}{\Delta/\sqrt{12}} = \frac{FSR/2\sqrt{2}}{FSR/2^n\sqrt{12}} = 2^{n-1}\sqrt{6} \equiv 6.02n + 1.76$

OVERSAMPLED ADC: $SNR = \frac{V_{eff}^2}{q^2/OS} = SNR_{Th} \cdot \sqrt{OS}$

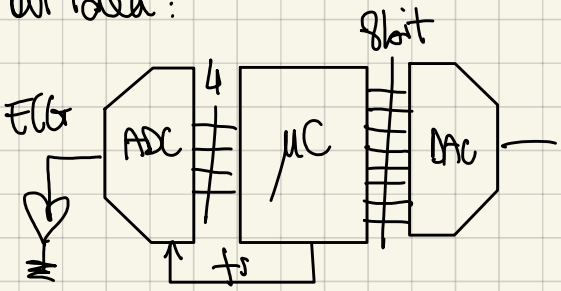
RESULT: we get an improvement of $\sqrt{2} = 3dB = 1/2$ bit every doubling of f_s!



$SNR_{ideal} = 6.02 \cdot 8 + 1.76 \approx 50dB$

$f_{MAX} \leq 250Hz$ $f_s = 2f_{MAX} = 500sps$ $T_s = 1/f_s = 2ms$

Our idea:

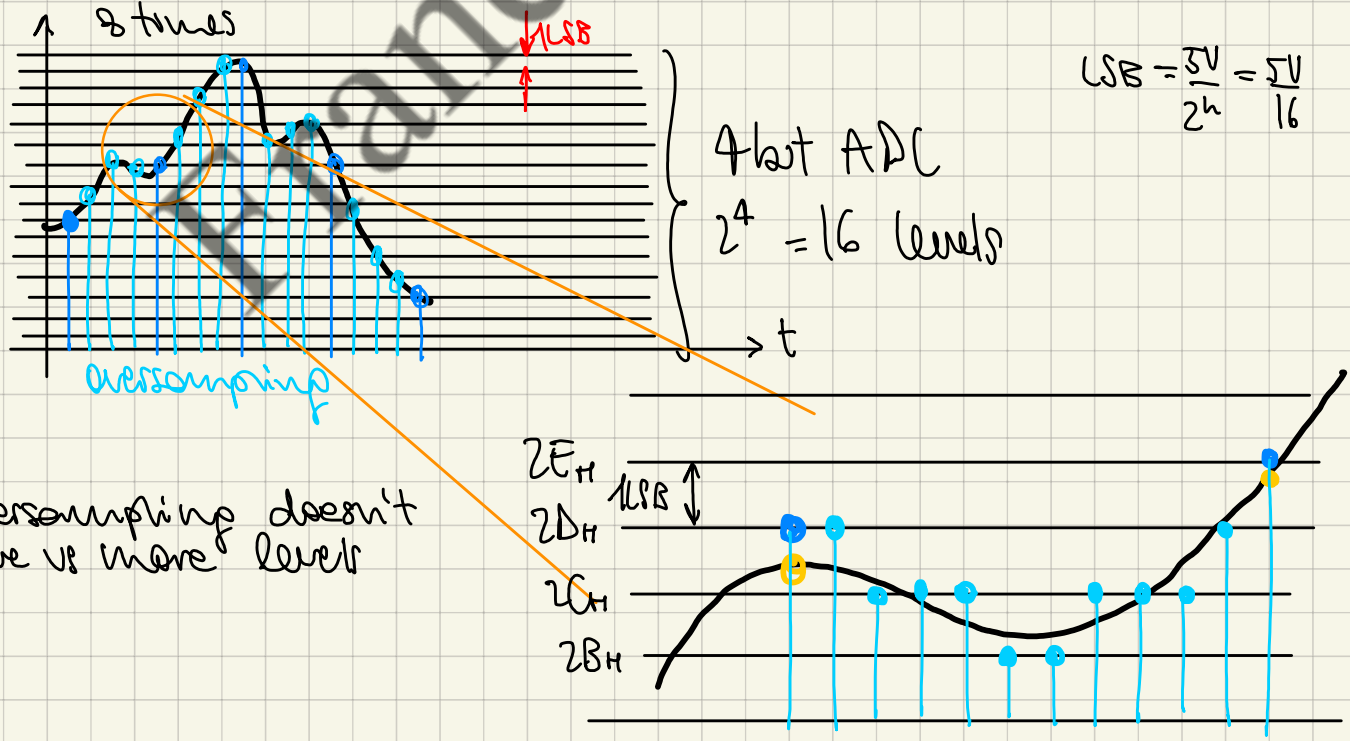


We want our µC to be able to provide 8 bit in terms of signal quality at the output

$f_{MAX} \leq 250Hz$ 8bit - 4bit = 4bit → improvement needed

Since we get a 1/2 bit improvement every doubling of f_s, if we want to achieve a 4bit improvement we need to implement 8 doublings:

$OS = \underbrace{2 \cdot 2 \cdot 2 \cdot \ldots \cdot 2}_{8 \text{ times}} = 2^8 = 256 \Rightarrow f_s = 2f_{MAX} \cdot OS = 256\,(500sps) = 128ksps$



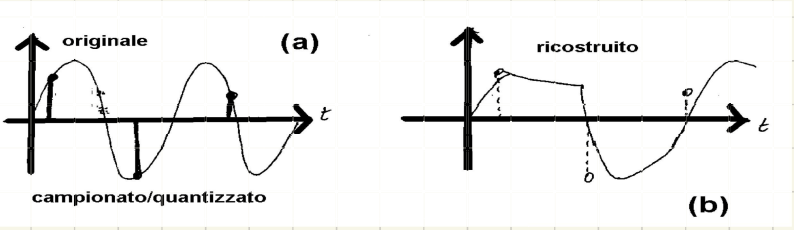
$LSB = \frac{5V}{2^n} = \frac{5V}{16}$

Oversampling doesn't give us more levels

How the µC can achieve an improvement in terms of its own resolution?

↓

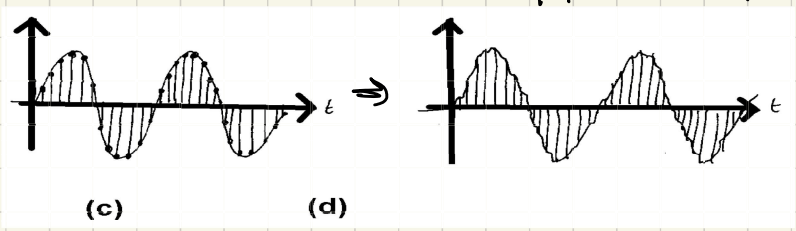
The improvement can be achieved by filtering



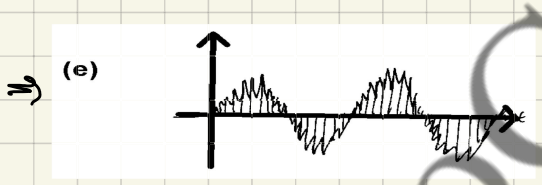
Using the Nyquist frequency and a LPF the quality of the reconstructed signal is poor

If we oversample the signal we have 2 possibilities:

- we use the same filter we'd use @ the Nyquist sampling rate

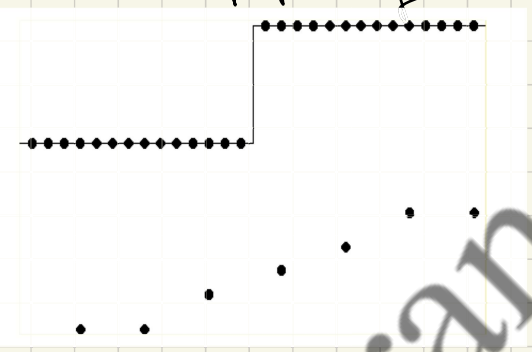


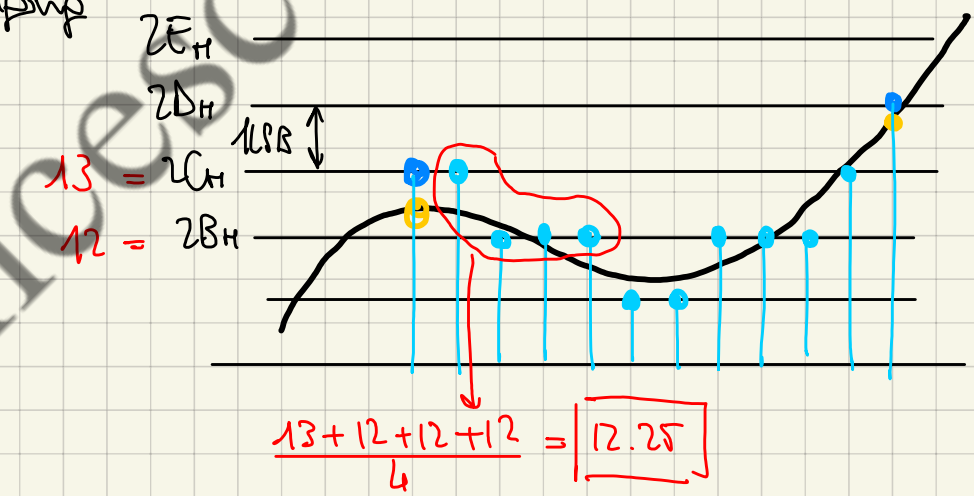
- we use a filter @ fs/2, which means that it has an higher bandpass so the τ is much shorter, this means that the filter is much faster in following all the new samples



DIGITAL FILTERING

The idea is performing averaging





After averaging our µC will have the possibility of having an arithmetics which is no more limited to the 8 bit

⇓

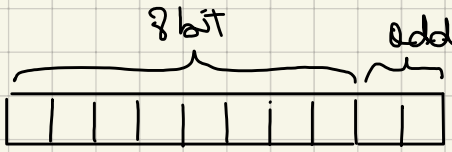
Averaging allows us to have an higher resolution in the µC's arithmetics

↓

This means that the accumulator of the µC should be able to perform sum btw different samples and then the division by the # of samples considered

NOTICE: The division by 4 means that we have to perform a shift leftwards of the accumulator register by 2 bit

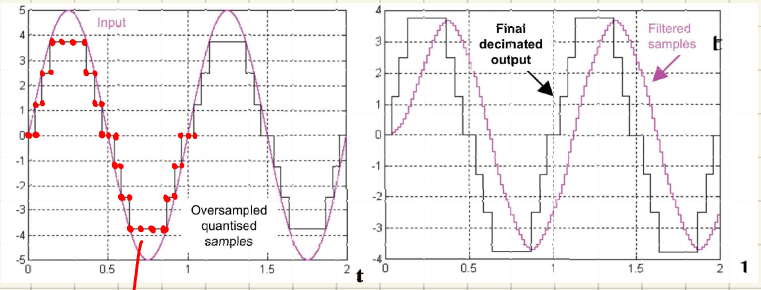
→ a shift on the left of 1bit corresponds to a division by 2



added resolution due to the decimal part resulting from the filtering action

⇒ Oversampling provides an improvement in the # of levels inside the µC thanks to the digital averaging of samples

EXAMPLE: $f_0 = 1kHz$ $OS = 16$ $f_s = 32sps$ $n = 3$ bit

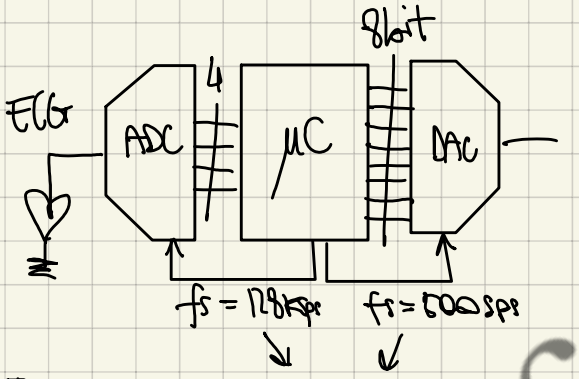


we acquire 16 samples more than what predicted by Shannon

even if we perform OS we're still limited by the resolution of the ADC (3 bit ADC)

Then filtering out the signal we basically perform an averaging which improves the resolution

Sooner or later the µC will provide the oversampled stream to a DAC

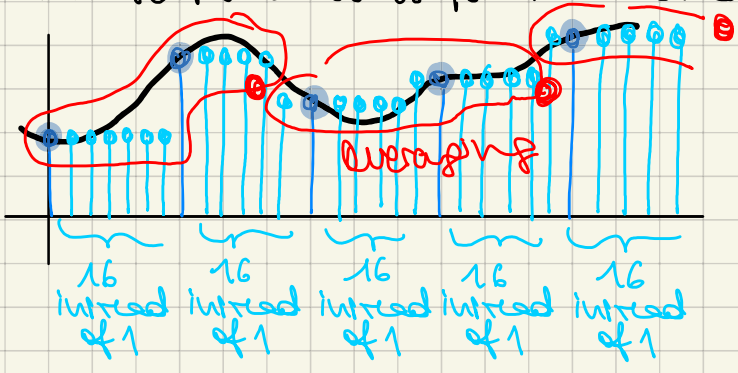


$f_{max} \leq 250Hz$

The two sampling freq.s differ one each other: the f_s @ which the µC makes the ADC operating is much higher than the Shannon limit in order to perform oversampling, while the DAC @ the output must operate @ $f_s = 2f_{max} = 500sps$

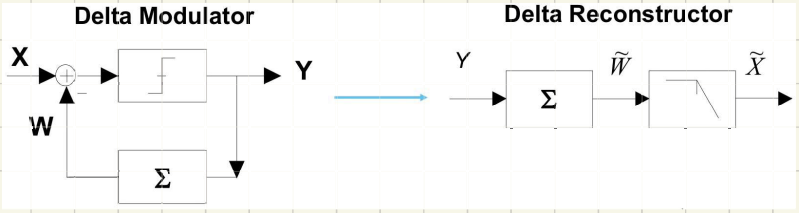
At the output DAC we can perform the so called **DECIMATION** which means that once we acquired a larger # of samples, but we're operating the ADC in a oversampling way, once we performed digital filtering, but we operated averaging, then we can simply store in our µC only one sample out of n

- select OS to provide the required bit improvement
- Then use the same OS for the decimation!!



OS = 16

DELTA MODULATION

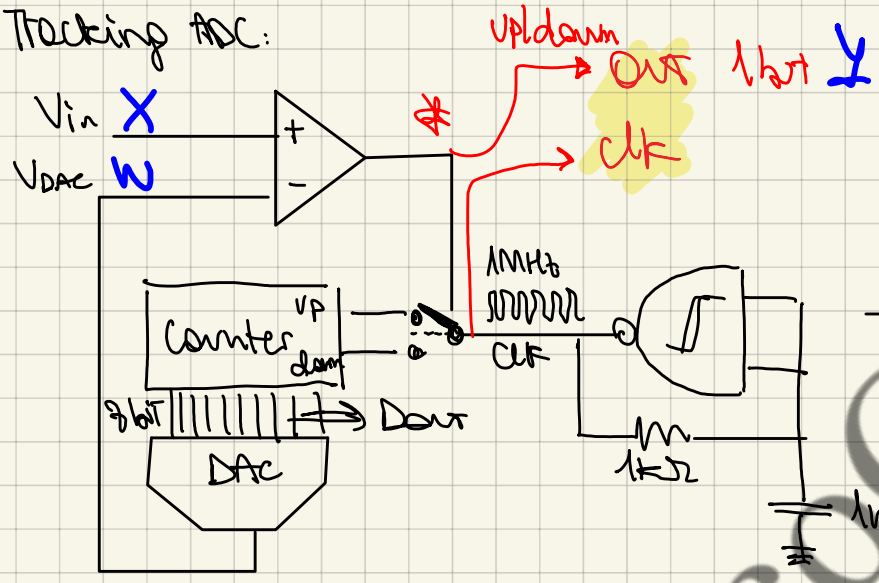


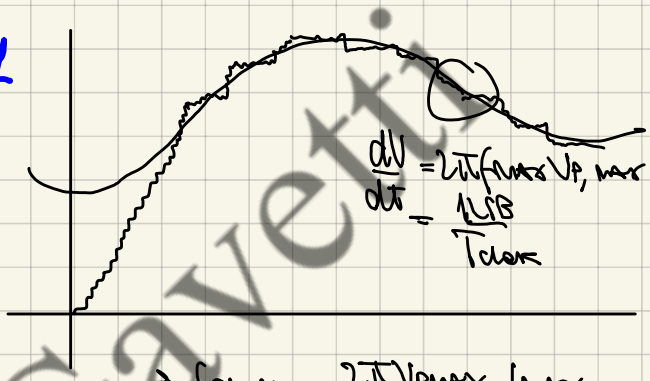
We already discussed the staircase ADC which is able to track the original signal and then we improved it by allowing the counter to go both up and down, discovering the so called TRACKING ADC

↓

The tracking ADC can also be described as a DELTA MODULATOR

Tracking ADC:





$$\Rightarrow f_{clock} = \frac{2\pi V_{pmax}}{1LSB} f_{max} = \frac{2\pi \, FSR/2}{FSR/2^n} f_{max} = \pi \, 2^n f_{max}$$

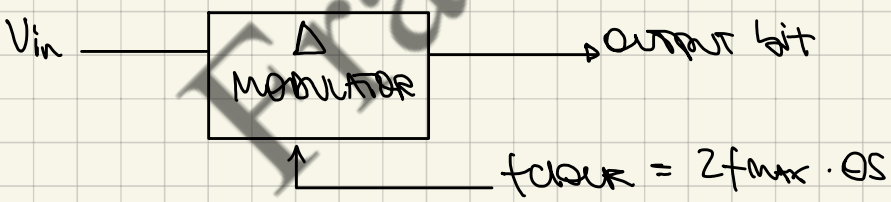
We get 1 DOUT every T_{clock}

↓

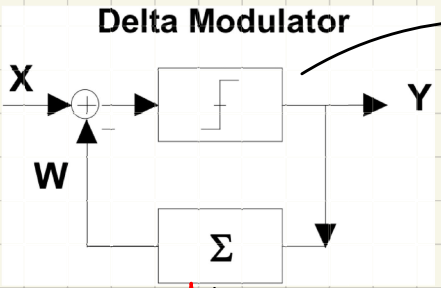
So instead of acquiring DOUT @ the output of the counter we can acquire it at the output of the comparator ✱

⇓

In this way this tracking ADC can be seen as 1bit oversampled ADC



What does the Δ modulator perform on the original signal?

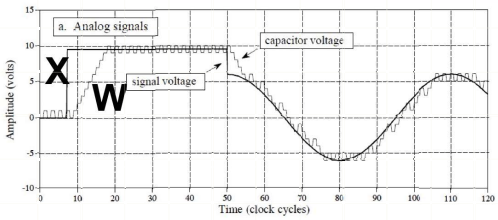


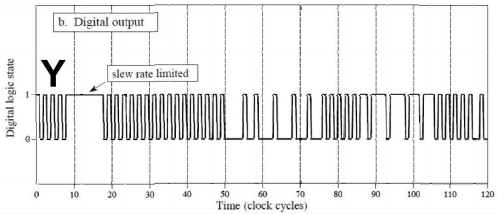
↳ it's a DISCRIMINATOR which behaves as COMPARATOR

- X = input signal
- W = reconstructed signal
- Y = output bits stream

↳ it's an ADDER that behaves as a COUNTER which is incrementing or decrementing @ every clock period and then followed by a DAC

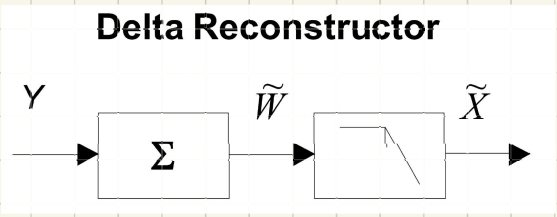
f_{max}



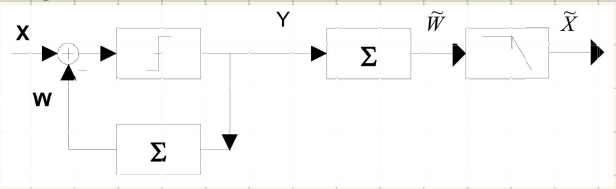


Notice: The Δ modulator can be considered as a 1st example of oversampling indeed we are not operating @ $f_s = 2f_{max}$ but @ $f_{clock} >> f_s = 2f_{max}$

Then we can take the output stream Y and provide it to a counter which is identical to the original one and finally we need a LPF to reconstruct the original signal

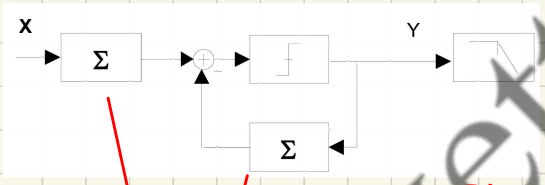


①



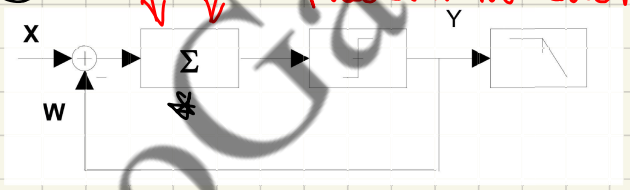
⇒

②



⇓

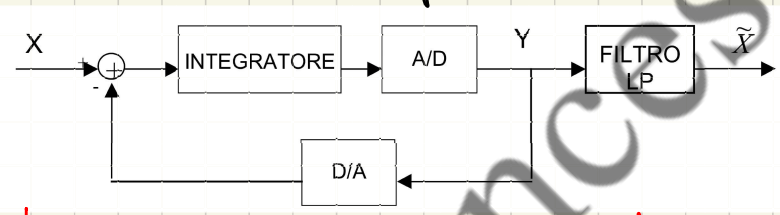
③



bc the two Σ converge to the same input differentiator node they can be placed in the same point w/in the loop

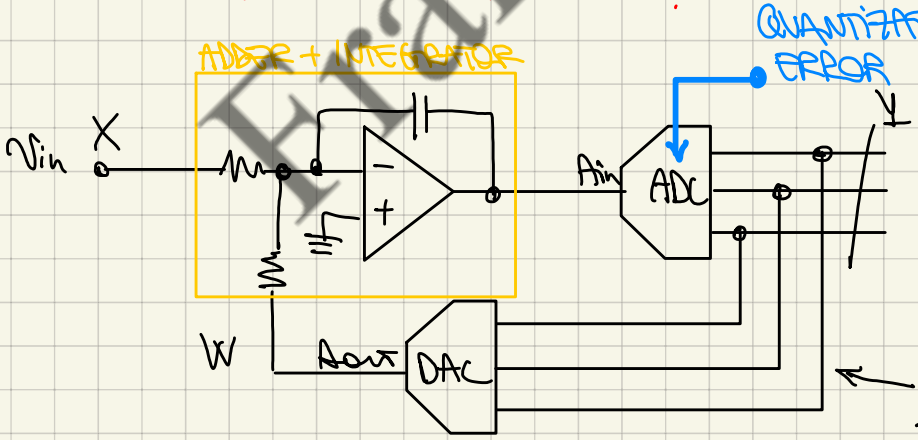
This configuration represents a ← ΣΔ MODULATOR

SIGMA-DELTA MODULATOR → it's an integrating network
* is no more the counter that we had at the beginning, but now it has a brand-new meaning:



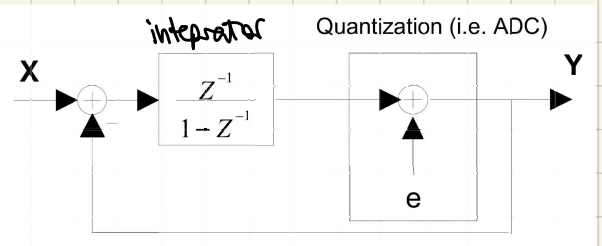
GENERAL ARCHITECTURE
The input signal is processed by a quantizer after the integration. The quantized output Y is reported to the input by means of a FB and subtracted from the input signal
Then the LPF performs the reconstruction of the original signal

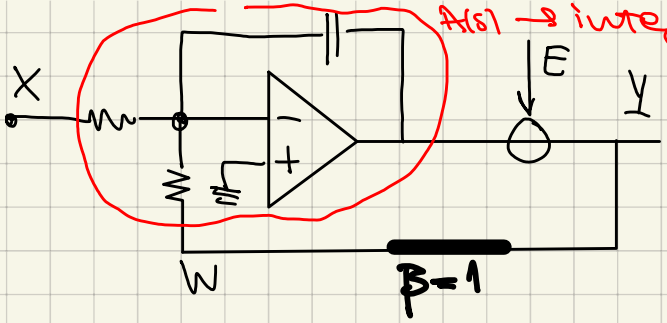
How can we model this schematic?

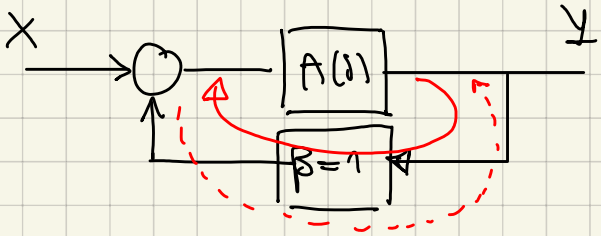


← here we have to introduce fclock

LINEAR MODELING:

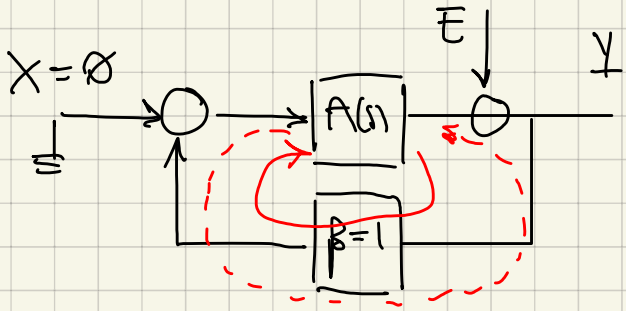






$$Y = X \cdot \frac{1}{\beta} = X$$

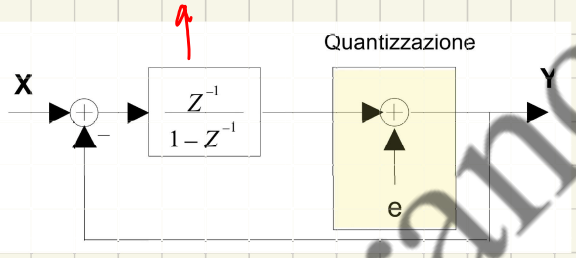
acts as a derivator



$$Y = E \frac{1}{A(s)\beta(s)} =$$

The loop contained in the circuit works as a modulator of the quantization noise but it reduces its presence w/in the band, shaping the spectrum to the higher freq.s
↳ this effect is known as NOISE SHAPING

Z-transform



$$Y(z) = X(z)\frac{1}{1+\frac{1-z^{-1}}{z^{-1}}} + e(z)\frac{1-z^{-1}}{1+\frac{1-z^{-1}}{z^{-1}}} =$$

$$= X(z)z^{-1} + e(z)(1-z^{-1})$$

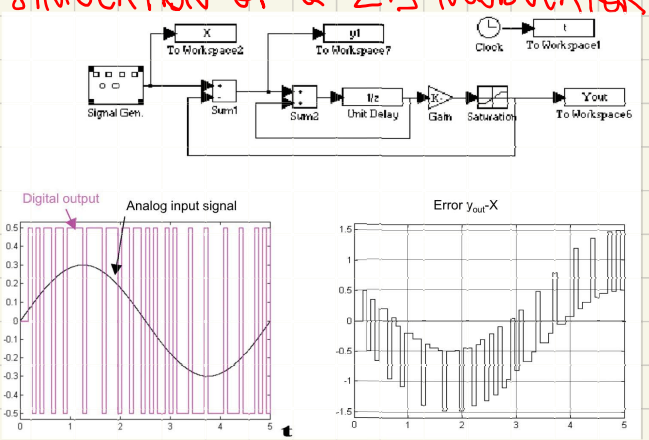
The average value of the output tracks the avg-value of the input signal

Eq is still present but it gets differentiated

1 clock pulse delayed

The noise shaping acts as $|\sin(\cdot)|^2$: $Q(f) = Q_0\left|1-e^{j2\pi fT}\right|^2 = 4Q_0\sin^2(\pi fT)$

SIMULATION of a ΣΔ MODULATOR



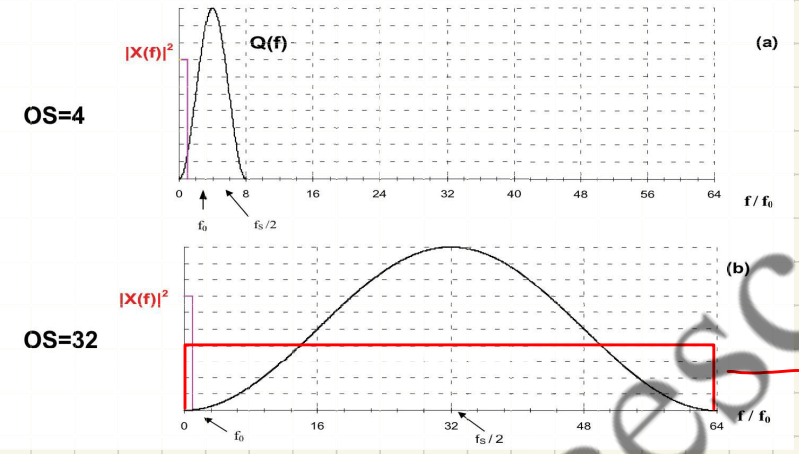
The error of the ΣΔ modulator is very high but it stays @ very high frequency

Why we may say that the ΣΔ modulator is eventually better than the classical oversampled ADC?

→ Due to this shaping here, the quantization error's spectrum is no longer flat as it was in the standard oversampling technique.
Thanks to the ΣΔ architecture, and so to the fact that 1st there's an integration and then there'll the ADC which introduce an error at the latter stage, we have an excellent NOISE SHAPING

$$Y(z) = X(z)\cdot z^{-1} + e(z)\cdot(1-z^{-1})$$

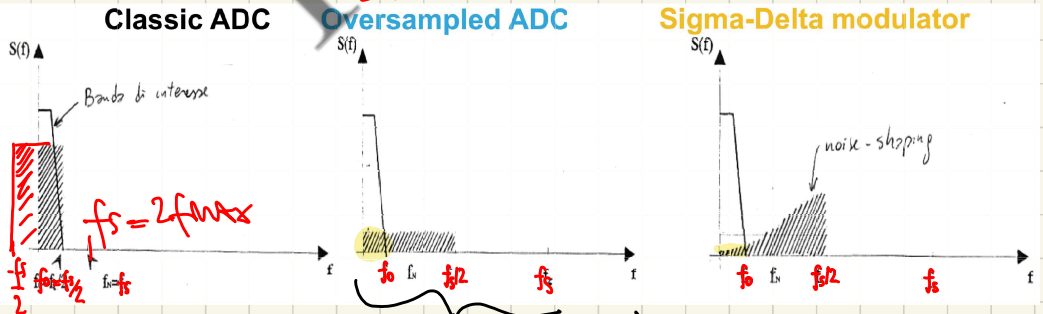
$$Q(f) = Q_0\left|1-e^{j2\pi fT}\right|^2 = 4Q_0\sin^2(\pi fT)$$



→ Eq is no longer flat btw 0 and fs as happens w/ the standard oversampling

from the pov of the quantization noise, the ΣΔ modulator improve the performance levels obtainable by the classical oversampling, thanks to the out-of-band noise shaping.
The noise is translated towards the high frequencies, increasing the oversampling factor.

ADVANTAGES OF ΣΔ MODULATION



In the oversampled ADC we use a higher fs, but we don't filter @ fs/2, we are smart and we filter @ fmax (fo)
In this way we "acquire" only a portion of the total Eq

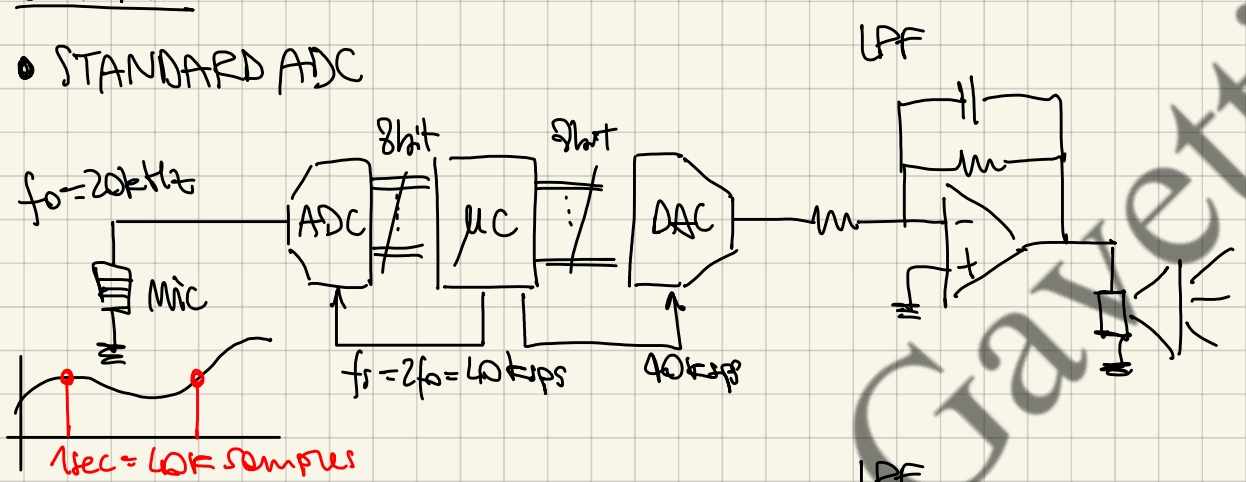
→ In fact, in this way we gain ½ bit every doubling of fs

(*) In the $\Sigma\Delta$ MODULATOR we get an even better result.
We sample @ an higher f_s w/r the standard ADC, as we do for the oversampled ADC we do not filter @ $f_s/2$, but we are smart and we filter @ f_o.
Here the advantage is that Eq gets a noise shaping so the portion of Eq we acquire is even lower w/r the amount we acquire w/ a standard oversampled ADC

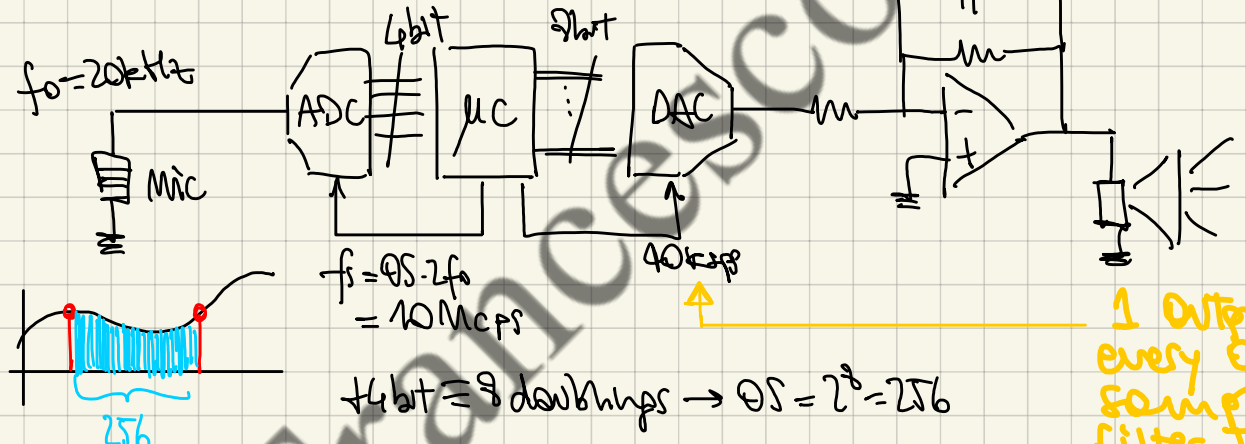
RESULT: Improvement of $2^3 = 9\,dB = 1.5$ bit every doubling of f_s (i.e. doubling of OS)

EXAMPLE:

- STANDARD ADC



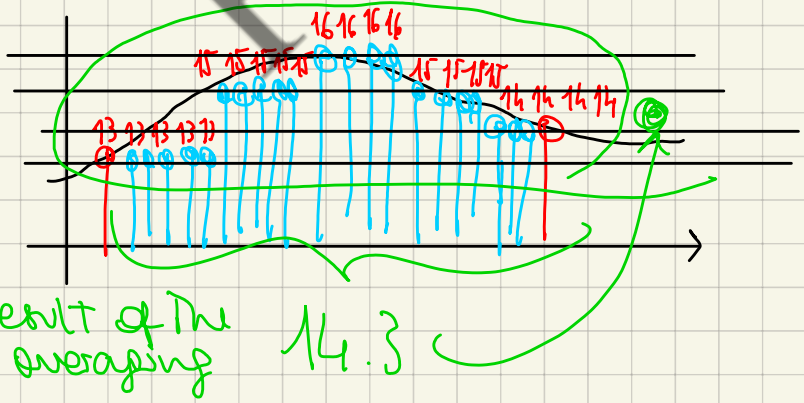
- OVERSAMPLED ADC



1 output sample every OS input samples and filter them w/ an average comparator

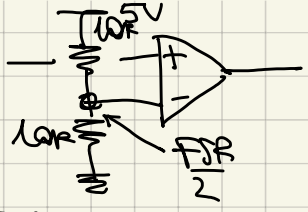
+4 bit = 8 doublings → OS = 2^8 = 256

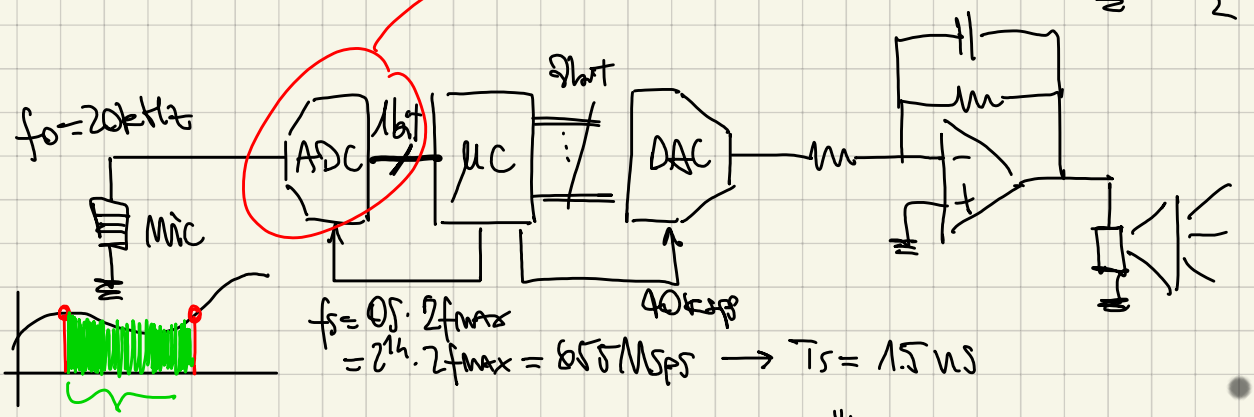
→ we have to use a larger memory !!!
↳ 10 MB



- EXTREME OVERSAMPLED ADC

A 1bit ADC is simply a comparator!! LF ⇒



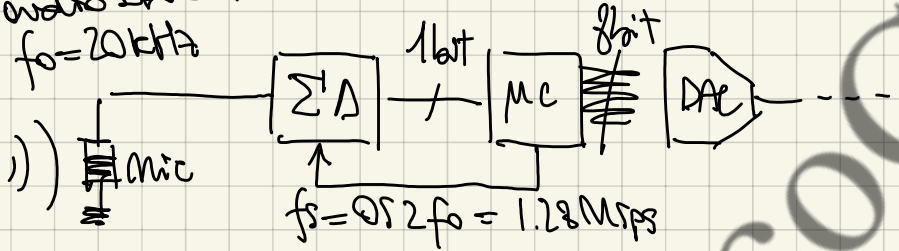


1sec = 655 Msamples +7 bit ≡ 14 doublings → OS = 2^{14} = 16384

↓

700 MB memory

- ΣΔ MODULATOR



+7bit ≡ 5 doublings → OS = 2^5 = 32

(+1.5 bit every doubling) → +7.5 bit actually

ΣΔ MODULATOR'S RESOLUTION

$$SNR_{dB} = 6.02\,(c + 1.5L) - 3.41$$

where c = bit of ΣΔ

$OS = 2^L$

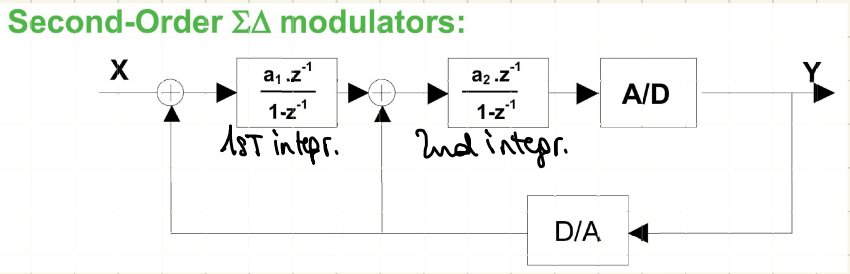
MAXIMUM EQUIVALENT # OF BITS

$$b = \frac{SNR + 3.41}{6.02}$$

⇒ Any filtering/processing that provides a number of bits higher than b is useless, since the extra-bits are redundant, since they simply describe the quantization error !

2nd ORDER Σ Δ MODULATOR

Second-Order ΣΔ modulators:

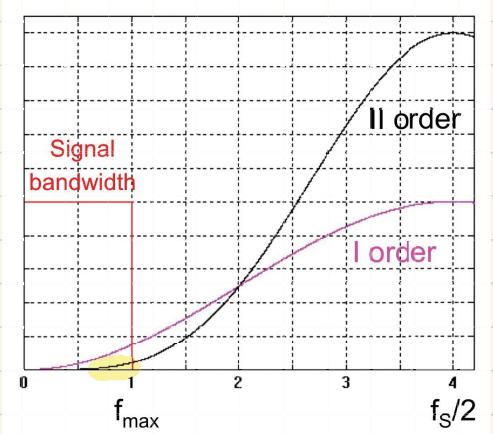


Notice: if the ADC is just 1 bit, so if it is a comparator, we don't need a DAC

$$Y(z) = X(z) \cdot z^{-2} + e(z) \cdot (1 - z^{-1})^2$$

→ the noise shaping acts no more as $\sin^2()$ but as $\sin^4()$:

$$Q(f) = 16\, Q_0 \sin^4(\pi f T)$$

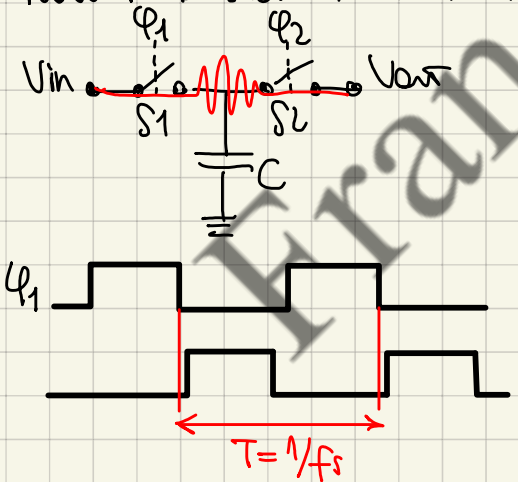


Filtering @ f_0, we collect even less eq w/ respect to the 1st order Σ'Δ modulator

RESULT: Improvement of 2?=? dB = 2.5 bit every doubling of f_S (i.e. doubling of OS)

How is a Σ'Δ modulator designed?

- HOW TO DESIGN A R IN A TIME-DISCRETE WORLD:



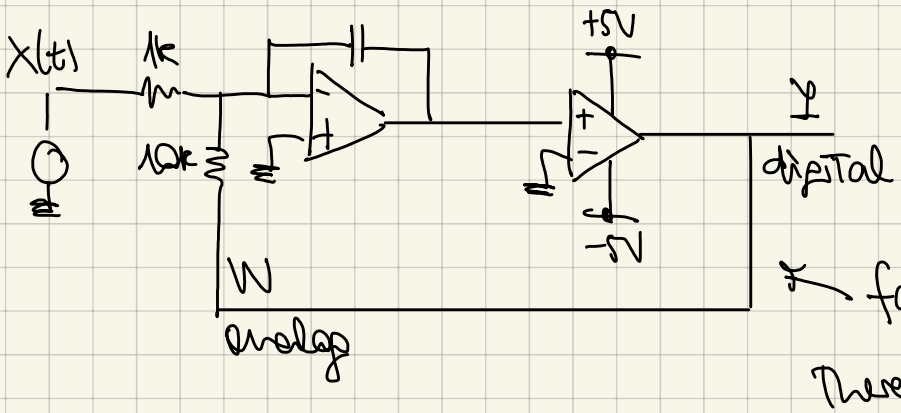
During each clock cycle we charge C to Vin and then we discharge it to Vout, so the charge transfer is:

$$Q = C(V_{in} - V_{out})$$

$$\rightarrow I = \frac{Q}{T} = f_S \cdot C (V_{IN} - V_{out})$$

In the continuous-time operation: $I = \frac{(V_{IN} - V_{out})}{R}$

$$\Rightarrow \boxed{R = \frac{1}{f_S \cdot C}}$$



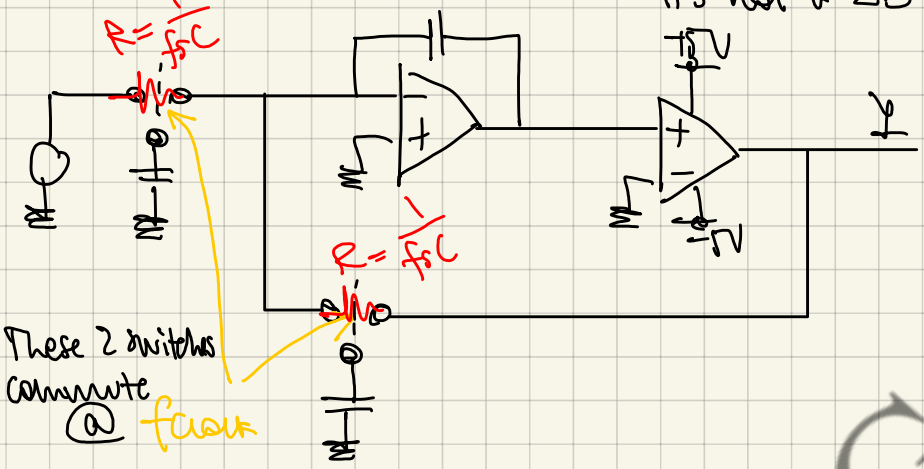
↗ fclock?

There's no fclock

⇓

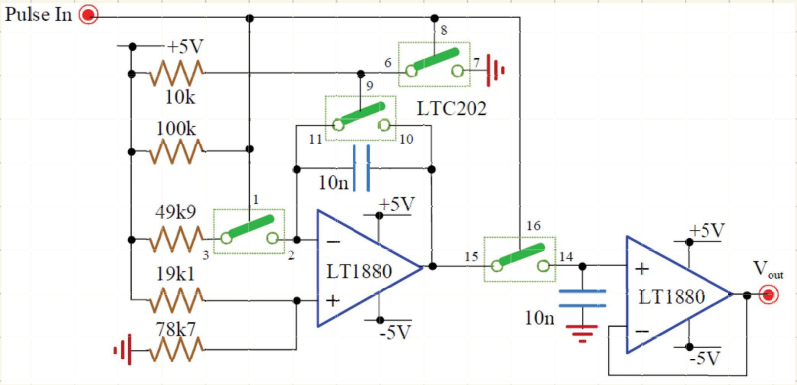
it's not a ΣΔ modulator, it's an amplifier

⇒ This is a ΣΔ modulator



ES16 - EXERCISES
15/12/2021

EX3



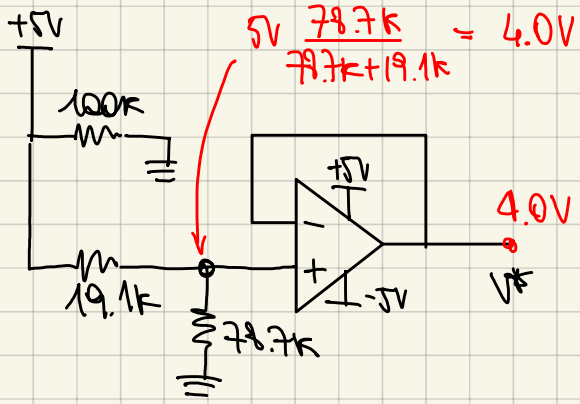
The LTC202 is a quad analog switch (closed when control pin is high)

The input is a pulse whose width Thigh is in 1ms ÷ 2ms range.

The Opamps have: $I_B < 1.5nA$ in the $-40° \div +85°C$ range

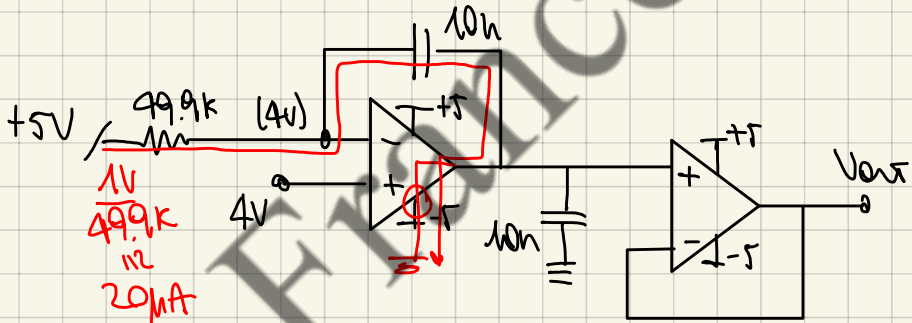
(A) Compute Vout as a function of Thigh

- Pulse in = low

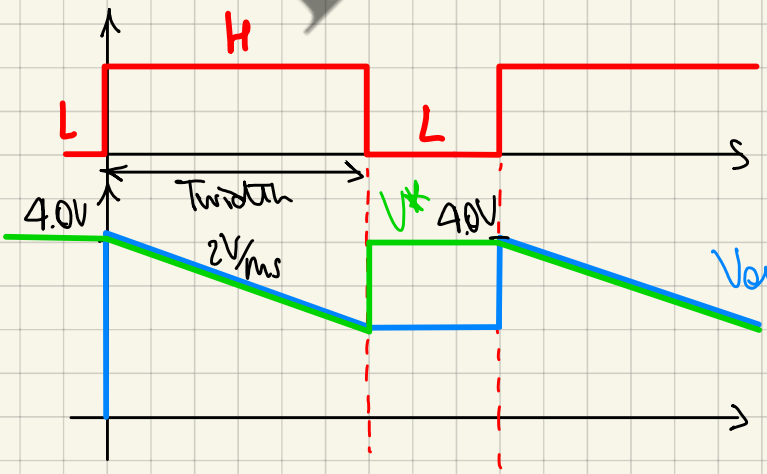


- Pulse in = high

integrator



$$\frac{dV}{dt} = \frac{i}{C} = \frac{20\mu A}{10nF} = 2kV/s \;\Big|\; = 2V/ms$$

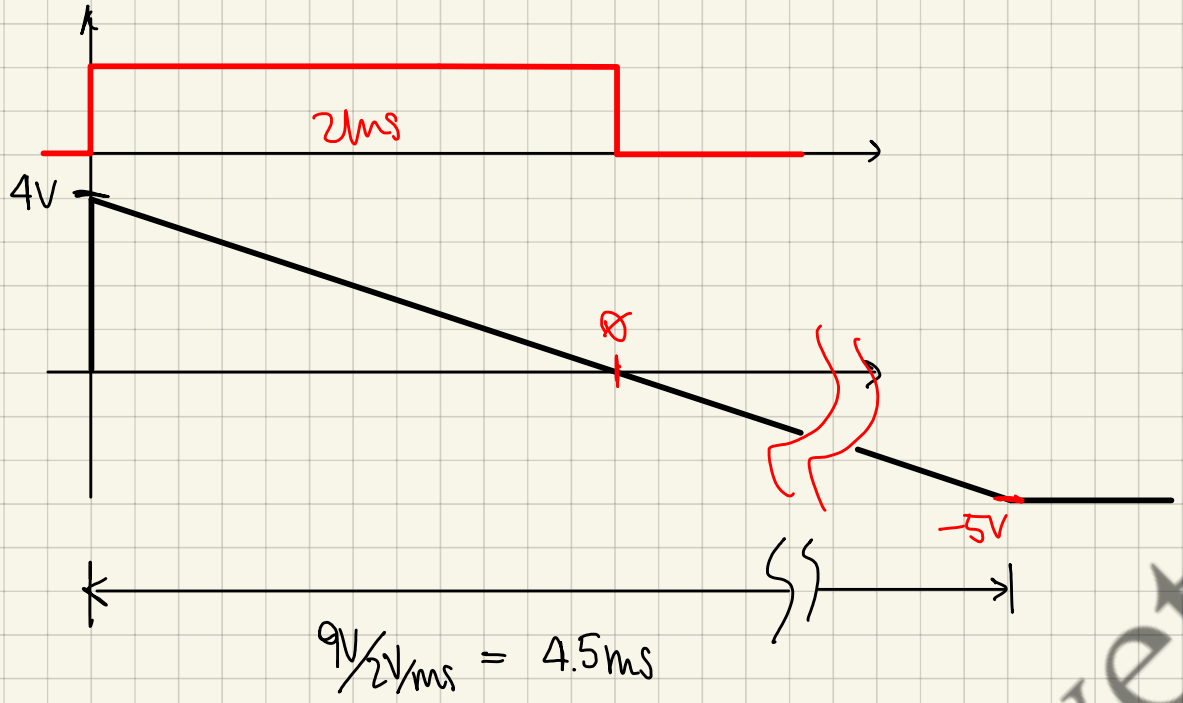


$$V_{out}(T_W) = 4V - \frac{2V}{ms} \cdot T_W$$

↓

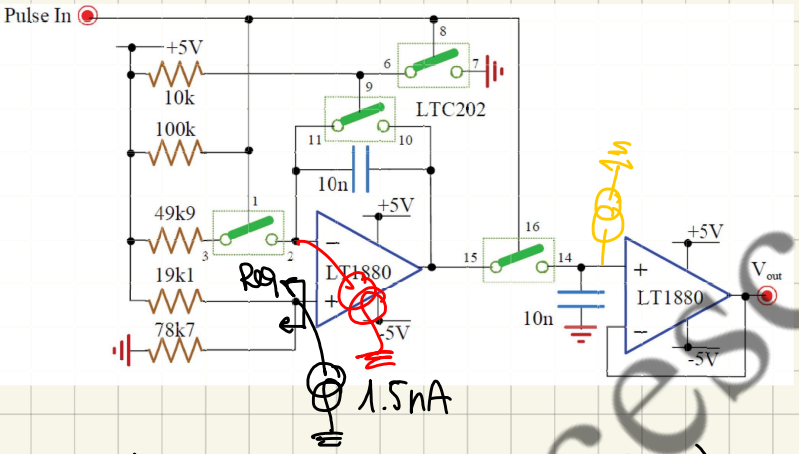
The output gives info about the duration of the pulse

There's a limit



$9V/2V/ms = 4.5ms$

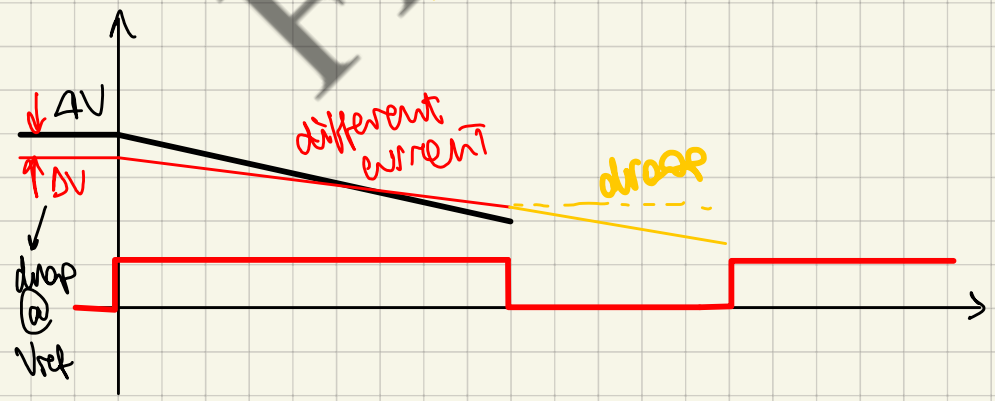
(B) Reckon the min Tsw that guarantees a precision of 1µs.



$\Delta V = 1.5nA\ Req = 1.5nA\ (19.1k \| 78.7k) = 15nA \cdot 15.4k = 23\mu V$

$i = 20\mu A - I_B =$

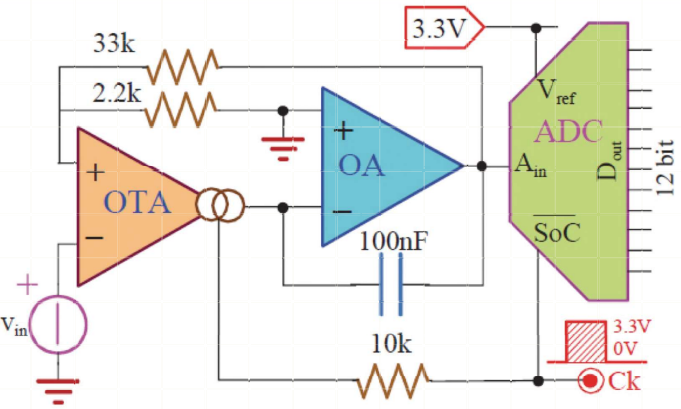
$droop = \frac{dV}{dt} = \frac{I_B}{C} = \frac{1.5nA}{10nF} = 0.15\frac{V}{s}$



ES16 - EXERCISES (3)

22/12/2021

EX12



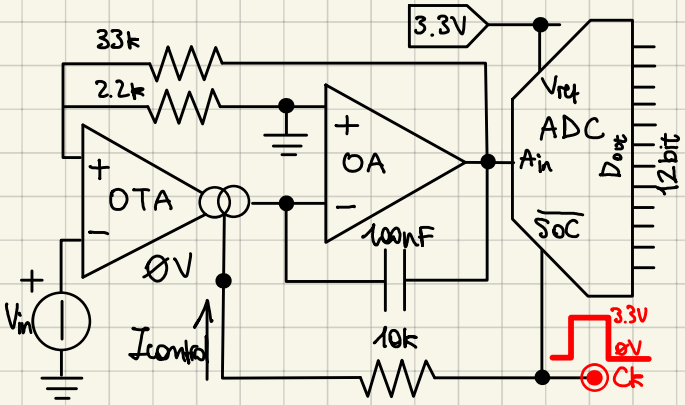
12 bit, 3.3V FSR ADC with $1mV_{rms}$ total input noise (at its A_{in} input). Ideal OpAmp. OTA control pin at 0V.

a) For a 5mV peak sinusoidal input with 5mV baseline, compute the SNR_{ideal}, SNR_{theor}, SNR_{real}, when considering just the $1mV_{rms}$ noise at the ADC input.

b) Compute the acquisition bandwidth and $t_{acquisition}$ for maximum FSR and ½LSB error.

c) Quote the static error in LSB due to I_B=1µA and Vos=2mV of the OpAmp.

Ⓐ



- Clk = 0

when Clk = 0, since control pin OTA = 0V, $I_{control} = 0A$

$G_m = \frac{I_{control}}{V_{in}} = 0 \rightarrow$ OTA is off

$\downarrow$

$I_{out} = 0$

This means that the voltage stored into the capacitor stays constant and so the circuit acts as a S&H stage in HOLD PHASE

- Clk = 3.3V → SAMPLING PHASE!

$I_{control} = \frac{3.3V}{10k} = 330\mu A \rightarrow$

$G_m = \frac{330\mu A}{25mV\ (@RT)} = 13.2\ mA/V$