Electronics systems notes

Lorenzo Pirovano
lorenzo2.pirovano@mail.polimi.it

February 1, 2026

Contents

Preface

These notes have been written in preparation for my oral exam in electronic systems in February 2026. They are provided as they are, in the hope that they may be useful. Mistakes may very well be present, and the reader is kindly invited to report them at my email address: `lorenzo2.pirovano@mail.polimi.it`

Please note that they also reflect a personal take on many of the subjects explained in the class, which may differ in form—albeit not in content—from what the professor is used to. I have attempted to simplify everything down to the essential wherever possible, following a personal philosophy of simple and detailed explanation, rather than the often bloated and confusing explanations I often have to deal with on a daily basis.

Disclamer:

These notes have been prepared in a hurry while studying for the oral exam. As such, no guarantee on the accuracy of their content can be made. Mistakes are likely present and everything the reader sees in here should be viewed under a lens of healty skepticism.

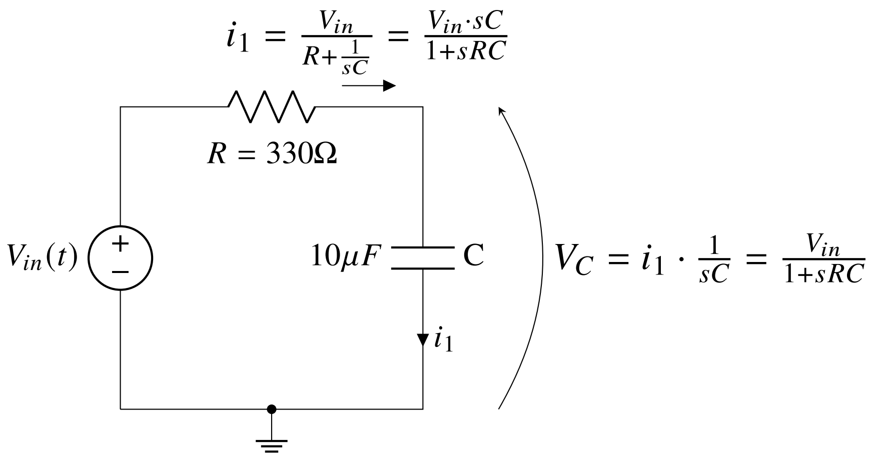
Chapter 1

Basics on electronics

1.1 Laplace analysis

Example 1: analytical approach

Laplace analysis can be used to analyze a *dynamical linear circuit* and compute its transfer function, defined as the ratio between the output and input. Let us consider, for instance, the circuit in figure 1.1. It consists of a very simple RC series, and it's easily the most simple example we could conceive. By considering the complex impedance of a capacitor as $Z_C = \frac{1}{sC}$, and writing Kirchhoff's laws, it's possible to work out the circuit's **transfer function**, defined as $G(s) = \frac{V_C}{V_{in}}$:



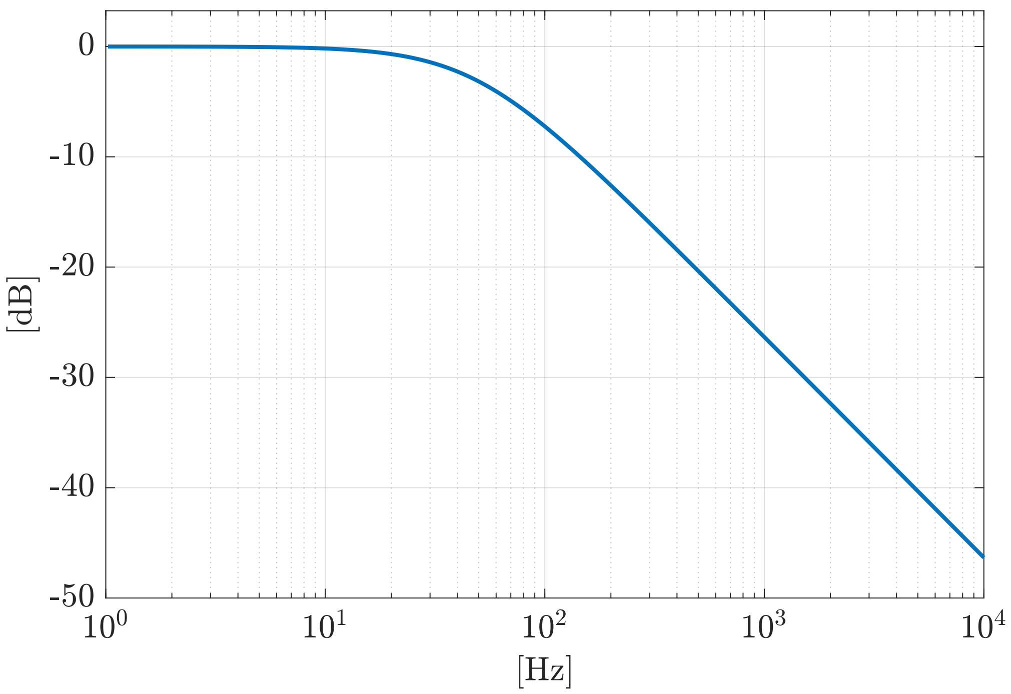
$i_1 = \frac{V_{in}}{R+\frac{1}{sC}} = \frac{V_{in} \cdot sC}{1+sRC}$
$R = 330\Omega$
$V_{in}(t)$
+
−
$10\mu F$
C
i_1
$V_C = i_1 \cdot \frac{1}{sC} = \frac{V_{in}}{1+sRC}$

Figure 1.1: Basic RC circuit.

We have computed $G(s) = \frac{1}{1+sRC}$. This transfer function shows a **pole** at $s = -\frac{1}{RC}$, whose frequency is $f_P = \frac{|s|}{2\pi} = \frac{1}{2\pi RC}$, and time constant $\tau = RC$. Keep in mind:
The pole is defined as the value of s for which the denominator of the transfer function is

zero.
The *physical meaning* of the pole is that the circuit's response to an external signal will contain some exponential terms like $e^{s \cdot t}$. This result could be obtained by applying the inverse Laplace transform of the transfer function multiplied by V_{in}/s, which is the circuit's response to a step signal of amplitude V_{in}.
If we apply Euler's formula, splitting the pole into its real and imaginary part: $s = \alpha + j\omega$, we find that $e^{s \cdot t} = e^{\alpha\, t} e^{j\omega\, t}$. In general, if $Re[s] < 0$, the circuit will be stable, as those exponential terms will be of the kind $e^{-t/\tau}$, which go to zero for infinite time. For this circuit, the solution to a step increase of V_{in} is: $V_C(t) = V_{in}(1 - e^{s \cdot t})$, which is zero for $t = 0$ and goes to V_{in} for $t \longrightarrow \infty$. Once the transfer function is known, the circuit's response can be fully determined, as long as it is a linear circuit.

It's common, in electronics, to represent these transfer functions in a so-called *"Bode plot"*, which is the plot of $20 \log_{10} |G(s = 2\pi j f)|$, on a logarithmic frequency axis. For this simple transfer function, this is the Bode plot:



At low frequencies, the capacitor may be regarded as an open circuit, with zero current. Therefore, the gain should be 1. As the frequency increases, the capacitor will act more and more as a short circuit, making the gain lower and lower as the output node cannot move.

Example 2: impedance partition:

A quick way to solve these circuits and find the transfer function is to use impedance partitions formulas. Since I use them all the time, I wanted to show it here, in its own example. This example circuit shows a C-R circuit, which is the opposite of the previous example:

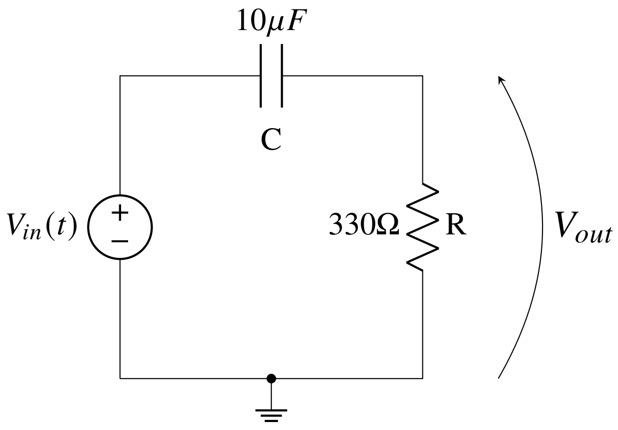
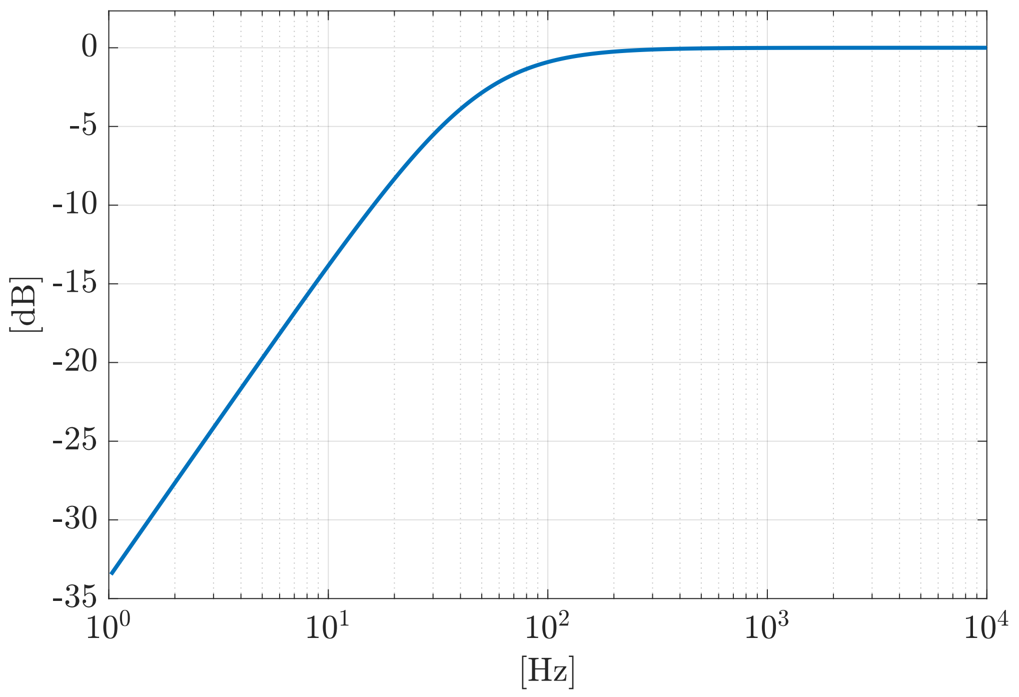


Figure 1.2: Basic CR circuit.

The transfer function can be obtained from the *voltage partition* formula between Z_C and R, which is always valid for any impedances in series with a voltage source: $V_{Z_j}/V_{gen} = Z_j/(\sum_i Z_i)$. For this circuit we can work out the following:

$$\frac{V_{out}}{V_{in}} = \frac{R}{R + Z_C} = \frac{R}{R + \frac{1}{sC}} = \frac{sCR}{1 + sCR}.$$

This transfer function shows a pole at $\tau_P = RC$, and a zero for $s = 0$, that is, for zero frequency, also known as DC. The zero is the value of s that makes the transfer function's numerator go to zero. In this case, it's just $s = 0$. By taking the limits for $s \longrightarrow 0$ and $s \longrightarrow \infty$ we can already tell that at low frequencies the circuit will strongly attenuate the input signal, while at high frequencies the gain will be almost 1. This is confirmed by shorting the capacitor (to simulate it's impedence going to 0 at high frequencies), or by removing it (to simulate its impedence being infinite at 0Hz). The Bode plot is the following, showing the expected trends:



The MATLAB code to produce these figures is found below:

```
%This program plots the bode plot of a transfer function in f domain:
Func = @(f)  (1)./(1+2*pi*f*1i*(330*10e-6)); %edit as needed.
fvector = logspace(0.01, 4, 1000); %1000 points, from 10^0.01Hz to 10^4 Hz
Func_vector = abs(Func(fvector)); %Evaluates abs(function)
Func_vector_db = 20*log(Func_vector)/log(10); %converts in dB
semilogx(fvector,Func_vector_db, 'LineWidth',2); %Plots
hold on;
ylim padded; %better scaling to view all the tranfer function
grid on;
box on;
xlabel("[Hz]");
ylabel("[dB]");
```

Example 3: asymptotic analysis

A further way to analyze these circuits and compute the transfer function is the so called ”Asyntotical analysis”. In this approach the circuit is solved twice: once at low frequency (hence, with the capacitor open) and once at high frequency (with the capacitor as a short). By doing this, the limits of the transfer function for $s \to 0$ and $s \to \infty$ are immediately found. Starting from those, and by computing the pole by inspection, the whole transfer function may be obtained. Let’s consider the following circuit:

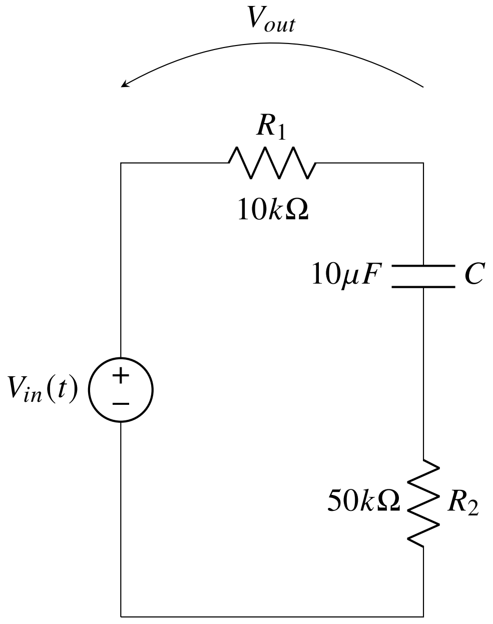


Figure 1.3: A more complex example of a circuit with resistors and a capacitor.

At DC, we may open the capacitor, resulting in no current flow, and therefore $V_{out} = 0$. At high frequencies, instead, the capacitor acts like a short circuit. Following the resistor partition formula, $V_{out} = V_{in} \cdot \frac{R_1}{R_1+R_2}$. The circuit is thus expected to have a zero for $s = 0$, and a pole at some frequency, resulting in a finite, non zero, high frequency transfer.

The pole may be computed by turning off the V_{in} generator and computing the resistance seen across C, which is just $R_1 + R_2$. The pole’s time constant is thus $\tau_P = C \cdot (R_1 + R_2) = 600\,ms$, and its frequency on the Bode plot is $f_p = \frac{1}{2\pi\tau_P} \approx 0.265\text{Hz}$.

The transfer function is thus expected to be $G(s) = \frac{s\cdot\tau_0}{1+s(R_1+R_2)C}$, where τ_0 can be found by using the high frequency gain: $\lim_{s\to\infty} G(s) = \frac{\tau_0}{(R_1+R_2)C} = \frac{R_1}{R_1+R_2}$, which implies $\tau_0 = CR_1$ and therefore $G(s) = \frac{sCR_1}{1+s(R_1+R_2)C}$.

Chapter 2

Negative feedback

2.1 Feedback theory

Example 1: real gain, loop gain, ideal gain

Let's consider the following reference circuit, which is known as a buffer. The opamp has gain $A_0 = 10^5$, which means its output will be $V_{out} = A_0(v^+ - v^-)$:

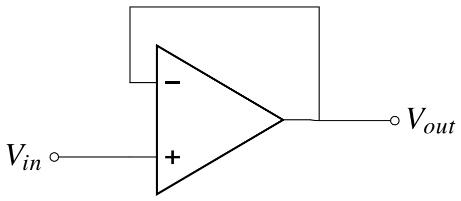


Figure 2.1: Opamp voltage buffer.

The amplifier is forcing $V_{out} = A_0(V_{in} - V_{out})$, which means $V_{out} = V_{in} \cdot \frac{A_0}{1+A_0}$. The *real gain* of this stage is thus $\frac{A_0}{1+A_0} = \frac{1}{1-\frac{1}{-A_0}}$. The quantity $-A_0$ is the *loop gain* of this stage, which is simply the ratio between a signal injected into the loop and the returning signal from the loop. If we imagine forcing a V_t voltage to the minus input of the opamp, the returning signal from the amplifier back to v^- would be $-A_0 \cdot V_t$, hence the loop gain is $-A_0$.

The *ideal gain* is defined, instead, as $G_{id} = \lim_{G_{loop} \to -\infty} G_{real}$, which for this stage is just one. This could be more easily computed by setting $v^- = v^+$, which is the so called *"Ideal opamp model"*[1].
Knowing the loop gain and the ideal gain, which are both easily computed by inspecting

[1]Whose equations are $v^+ = v^-$, $i^+ = i^- = 0$.

the circuit, the real gain can be computed as:

$$G_{real} = \frac{G_{id}}{1 - \frac{1}{G_{loop}}} = \frac{G_{id} \cdot G_{loop}}{1 - G_{loop}}. \tag{2.1}$$

One interesting property of equation 2.1 is the reduced effect of variations of G_{loop} on G_{real}. By computing $\frac{dG_{real}}{dG_{loop}} = \frac{d}{dG_{loop}}(\frac{G_{id} \cdot G_{loop}}{1-G_{loop}}) = G_{id}\frac{-(1-G_{loop})-G_{loop}}{(1-G_{loop})^2} = \frac{-1}{(1-G_{loop})^2}$, we can notice that

$$\frac{dG_{real}}{G_{real}} = \ldots = \frac{-dG_{loop}}{G_{loop}(1 - G_{loop})} \approx \frac{dG_{loop}}{G_{loop}} \cdot \frac{1}{G_{loop}} \tag{2.2}$$

Which means that a variation of 50% of G_{loop}, that is, $dG_{loop}/G_{loop} = -0.5$, if $G_{loop} = 10^5$, produces a change of G_{real} of only $dG_{real}/G_{real} = 0.5/10^5 \approx 0.0005\%$, which is negligible. In other words, as long as the loop gain is high enough, its variations won’t be reflected in the real gain, paving the way for very precise circuits, even if the components that make them up, especially the opamps, are not precise.

2.2 Input and output resistances for circuits with feedback:

Example 2: input and output resistances of a buffer

Let’s consider the circuit in figure 2.2, which is the same as the circuit in figure 2.1, but it includes certain parasitic resistances which are present in real opamps: $[A_0 = 10^5, R_1 = 1\ M\Omega, R_2 = 100\ \Omega]$

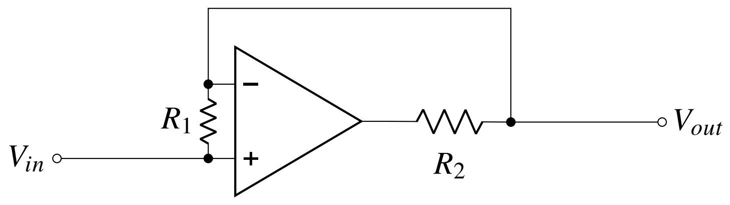


Figure 2.2: Opamp voltage buffer with parasitic R_1 and R_2.

The loop gain of this stage is affected by the parasitic resistances, and it’s now $G_{loop} = -A_0\frac{R_1}{R_1+R_2} \approx -99990$. This result can be achieved by applying the voltage partition formula, starting with a test signal across R_1, which is then amplified by the opamp, and partitioned back into R_1.

The output resistance of the stage is affected by feedback. If feedback were not present, the resistance seen would be $R_{out}^{(0)} = R_2//R_1$. If we apply a test current to the output, a fraction of it will flow into R_2, generating a positive voltage signal at v^-. This will swing down the output of the opamp by a large amount (since its gain is high), thus making more of the injected current flow into R_2, and less into R_1. This behavior indicates that the feedback attempts to keep the V_{out} constant despite the injected current, thus indicating a sort of *"virtual ground"*. When this happens, the resistance seen at the output can be computed as:

$$R = \frac{R^{(0)}}{1 - G_{loop}} \tag{2.3}$$

For our example, this results in $R_{out} \approx \frac{100\Omega}{99991} \approx 0.001\Omega$, which is very low and allows this stage to act as an ideal voltage generator.

The input resistance is also affected by feedback. If feedback were not present, an input generator would simply see $R_{in}^{(0)} = R_1 + R_2$. Now, if we imagine injecting a current into the stage from V_{in}, this current will flow into R_1, resulting in an initial increase of v_+. This, in turns, makes the output of the opamp rise significantly. This increase is fed back by R_2 to v^-, where it activates a current flowing down that attempts to counter the current initially injected into R_1. Ideally, no current would flow into R_1 and the input resistance would be infinite. We can compute the actual input impedance as:

$$R = R^{(0)}(1 - G_{loop}) \tag{2.4}$$

Which, for this example, means $R_{in} = (1M\Omega + 100\Omega) \cdot (99991) \approx 100\ G\Omega$. This value is very high, which allows this stage to read a voltage signal without absorbing a current from the source, making it ideal for the role of a buffer. Note how equations 2.3 and 2.4 are somewhat opposite, and a good understanding of the circuit's feedback is needed to select the correct one.

2.3 Frequency respose of a feedback circuit

Equation 2.1 is also valid in the s domain. By remembering that the poles are the values of s for which the denominator of the transfer function is zero, we can compute the closed loop poles by simply solving

$$G_{loop}(s) = 1 \tag{2.5}$$

The zeros, instead, are usually the same as those of G_{id}, but in general they are still obtained by equation 2.1. Let's compute the closed loop pole of the circuit in figure 2.1, assuming that the opamp's response features a pole at $f_0 = 100$ Hz, whose time constant is $\tau_0 = \frac{1}{2\pi f_0} \approx 1.6\ ms$. This means that the opamp's transfer will be $V_{out} = A(s) \cdot (v^+ - v^-)$, with $A(s) = \frac{A_0}{1+s\tau_0}$. The loop gain will be $G_{loop} = -\frac{A_0}{1+s\tau_0}$. By solving equation 2.5 we can find the closed loop pole of the system, which will be $s = -\frac{A_0+1}{\tau_0} \approx -\frac{A_0}{\tau_0}$. Its frequency on the Bode plot will be $f_p = \frac{|s|}{2\pi} = \frac{A_0}{2\pi\tau_0} = A_0 \cdot f_0 \approx 10$ MHz. As you can see, feedback has increased the bandwidth of the stage from just 100 Hz in open loop, to 10 MHz at closed loop, an increase of a factor G_{loop}. These are the Bode plots of G_{loop} and G_{real};

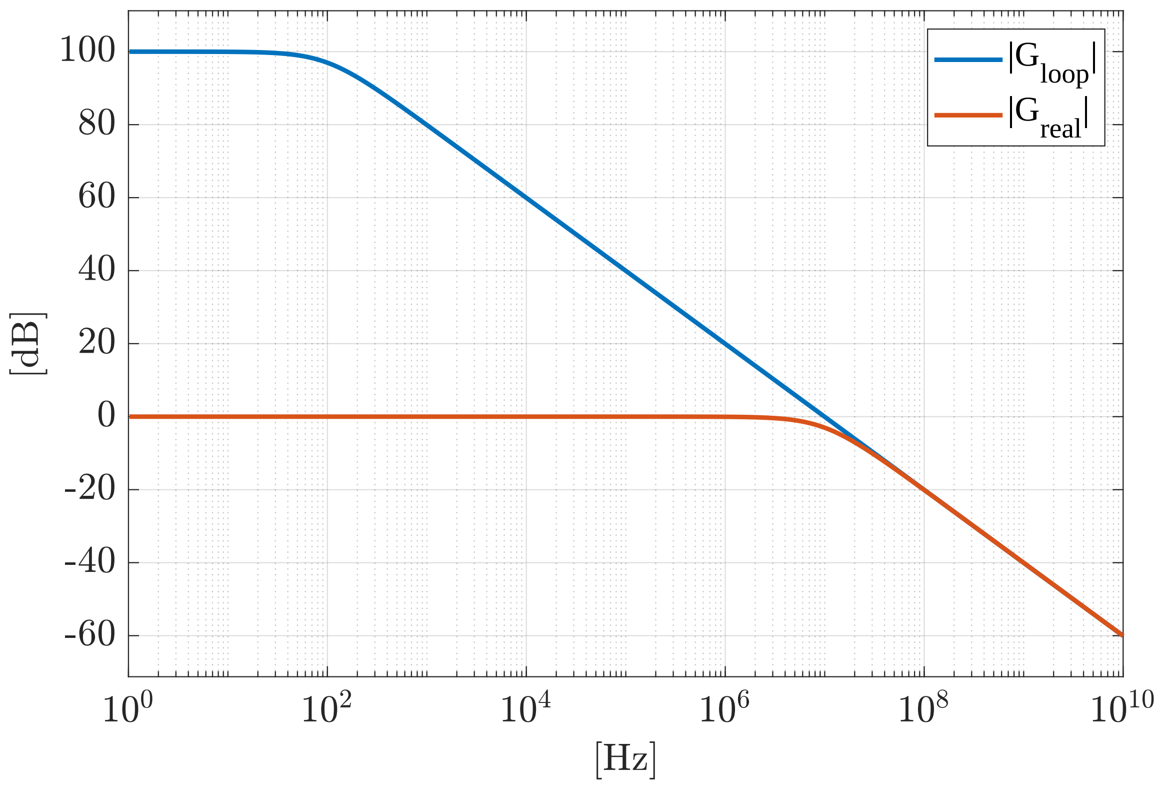
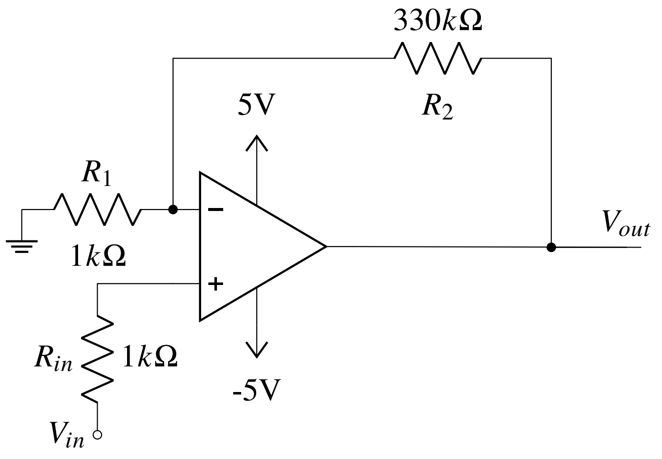


Figure 2.3: Bode plot of $|G_{loop}|$ and $|G_{real}|$.

As you can see, as long as $|G_{loop}| > 1$ [0dB], the real gain will be almost one. Only once G_{loop} drops below 0dB we'll begin seeing changes in G_{real}. Ultimatly, this is the reason for the increase in bandwidth achieved by feedback.

2.4 Trimming away the offset

Asymmetries in the differential stage within the opamp make it so that its output follows $V_{out} = A_0(v^+ - v^- \pm V_{os})$, where V_{os} is known as the ”offset” of the opamp. This offset will result in a net output voltage different from zero, even when no signal is being applied. For instance, let’s consider the non inverting stage in 2.4:



$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1+R_2}$$

$$\text{GBWP} = G_{loop}(0) \cdot f_0 \approx 30.21\text{KHz}$$

$$G_{id} = 1 + R_2/R_1 = 331$$

Figure 2.4: Opamp non-inverting stage.

The offset will result in an output drift of $V_{out} = (1 + R_2/R_1)V_{os}$, which, for $V_{os} = \pm 5\ mV$ will result in $V_{out} = \pm 1.655\ V$. That’s extremly undesirable, and there are ways to prevent it. One possible solution is the following:

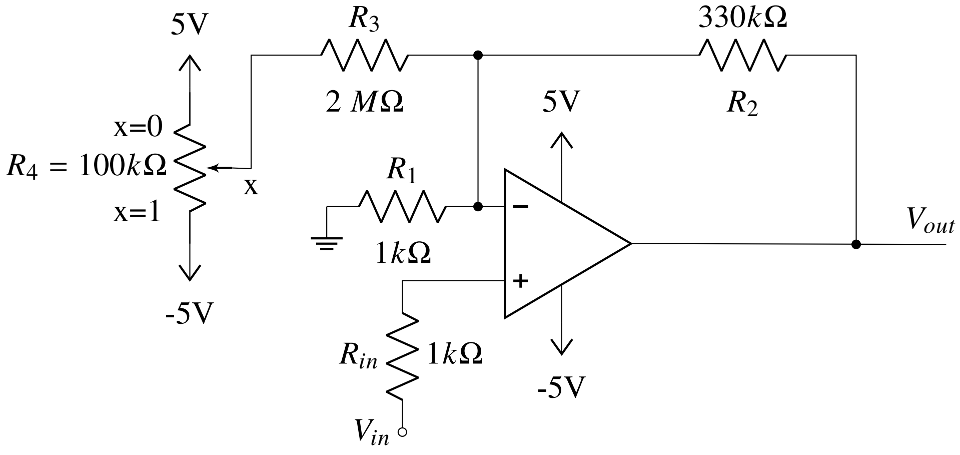


Figure 2.5: Opamp non-inverting stage with offset nulling.

The idea of this circuit is to use the R_3 network to induce a current into R_2, which may have either sign, to slightly nudge the output voltage and allow it to be zero even when offset is present. If $R_3 >> R_1$, the ideal gain is not affected. Making the computations, we find:

$$V_{out} = V_{in}(1 + \frac{R_2}{R_1}) \pm V_{os}(1 + \frac{R_2}{R_1}) - 5V\frac{R_2(2x-1)}{R_4x^2 - R_4x - R_3}$$

Which allows the offset to be compensated by choosing an appropriate value of x. In practice, this is done manually, by trimming R_4 with no signal applied until $V_{out} = 0$. Other stages may be compensated in a similar fashion, and the datasheets for the opamps often provide the necessary topologies.

2.5 Single polarity power supply

All the stages seen so far have been supplied using $\pm Vdd$, which, for a portable system, requires two different batteries connected in series. This is done in many circuits, especially in musical systems, but there are also way to avoid this. Note that a naive solution like figure 2.6 isn't always appropriate, as it can only amplify input signals $V_{in} > 0$. Any negative input signal, even just $1mV$, would make the output saturate at $0V$ and prevent the amplification of the signal.

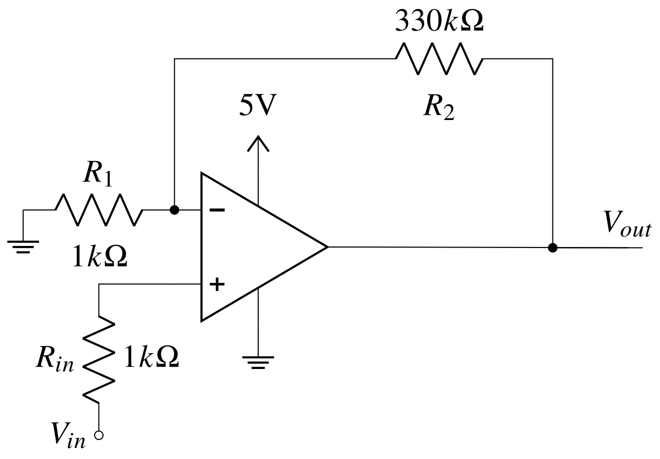


Figure 2.6: Opamp non-inverting stage with 5V single power supply.

If we wish to amplify signals with both polarities, we need to decouple the input signal using a capacitor which shorts at high frequency, but still allows different bias points of the amplifier and the signal in DC. If the signal is biased at 0V, we may want to bias the amplifier at $Vdd/2$ to allow the greatest possible range of inputs. This is a possible solution:

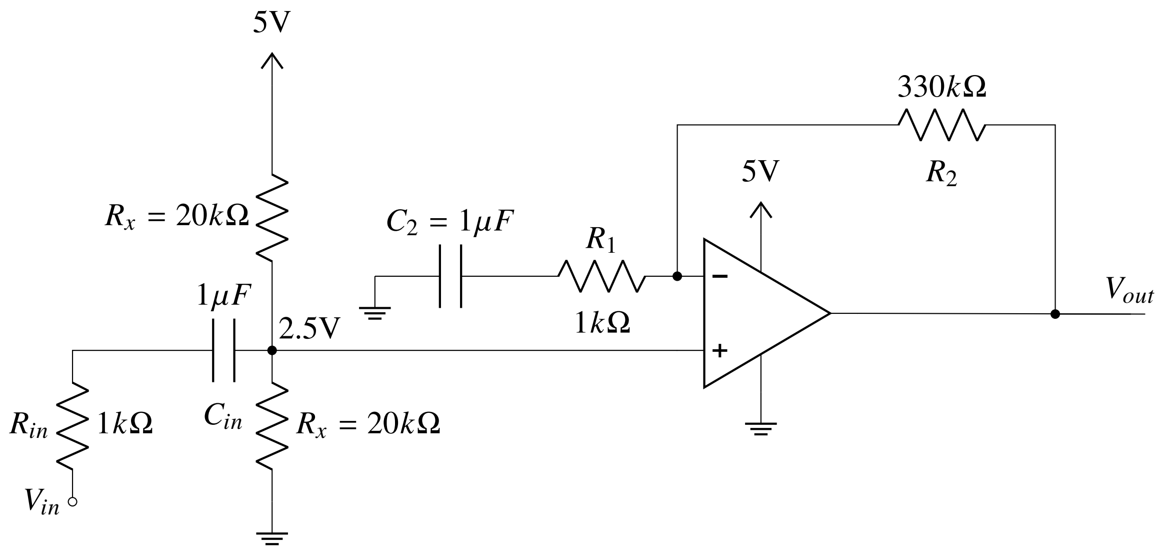


Figure 2.7: Opamp non-inverting stage with 5V single power supply and 2.5V bias.

The input partition sets the bias point of both input and output at 2.5V, while C_{in} allows to connect an input which is referred to 0V. C_2 is needed to reduce the DC gain down to 1, which prevents the amplifier from amplifying the 2.5V bias of the stage. At higher frequencies, C_2 will short and the gain won't be affected.
C_{in} will introduce a zero in the origin and a pole with $\tau_1 = C_{in}(R_{in} + R_x/2)$, while C_2 will introduce both a pole and a zero, with the pole at $\tau_2 = C_2 R_1$. Its zero can be found by the constrains of the low frequency gain being 1, and the high frequency gain being $1 + R_2/R_3$ (looking at just the transfer from v^+ to V_{out}). This discussion highlights how the various components should be sized depending on the frequency of the input signal.

2.6 Limitations due to output current

2.6.1 With a resistive load

When the output of the opamp drives a resistive load, the V_{out} value determines a current $i_{out} = V_{out}/R_L$ flowing in the load, which must be provided by the opamp. Let's consider the simple example of a voltage buffer driving a resistive load:

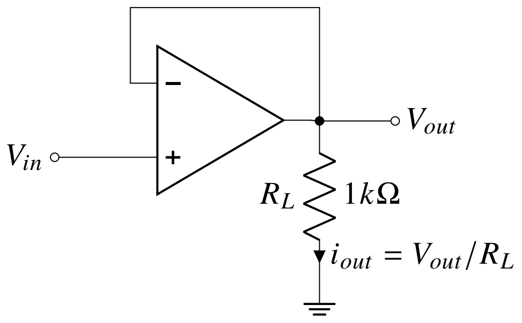


Figure 2.8: Opamp voltage buffer with resistive load

If the output of the amplifier has a limitation in current, let' s call it i_{max}, the corresponding maximum output swing will be $\pm i_{max}R_L$. It is the designer's responsibility to select an opamp with an appropriate i_{max} for the desired load and swing. Given a ΔV maximum desired swing, and a minimum load R_L, one must select $i_{max} \geq \frac{\Delta V}{R_L}$. If we imagine a desired swing of 2V with $R_L = 1k\Omega$ for instance, that requires $i_{max} \geq 2mA$.

2.6.2 With a capacitive load

A capacitor follows the relation $i_C = C\frac{dV_C}{dt}$. This means that the limitation won't be set by the static value of V_{out}, but by how quickly the output node moves. Let's consider the circuit below:

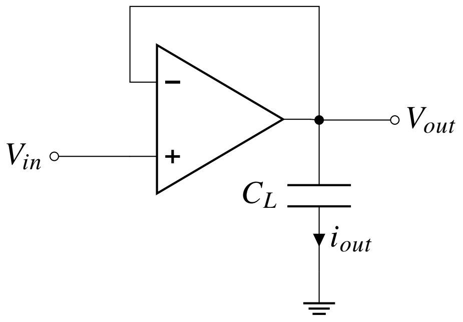


Figure 2.9: Opamp voltage buffer with capacitive load.

If we assume $V_{in} = V_{out} = A_0 \sin(2\pi ft)$, the current flowing out of the opamp will be $i_{out} = C\frac{dV_{out}}{dt}C_L A_0 2\pi f \cos(2\pi ft)$, which is greatest at $t = 0$, and it's $i_{out}^{max} = C_L A_0 2\pi f$.

Given a maximum output current i_{max}, the maximum achievable frequency is $f_{max} = \frac{i_{max}}{2\pi A_0 C_L}$, which, if the sinusoid has a full swing of V_{dd} -the maximum possible- is known as FPBW (Full Power BandWidth).

$$FPBW = \frac{i_{max}}{2\pi V_{dd} C_L}. \tag{2.6}$$

2.7 Slew rate limitation and FPBW

The SR (Slew Rate, rapporto di Lumaca), dell'opamp, is defined as the maximum derivative of the output node that the opamp can handle. Upon reaching this limit, the response of the opamp (in time) will be a straight line with slope equal to the SR. Remember:

$$SR = \frac{dV_{out}}{dt}|_{max}.$$

If we assume $V_{out} = V_{dd}\sin(2\pi f t)$, the maximum derivative of the output signal will be $\frac{dV_{out}}{dt}|_{max} = V_{dd} 2\pi f = SR$, leading to a maximum allowed frequency of:

$$FPBW = \frac{SR}{2\pi V_{dd}} \tag{2.7}$$

Both equation 2.6 and equation 2.7 impose a limit on the maximum bandwidth of the stage, which is ultimately limited by the lowest of the two. They are responsible, respectively, for the so called *external SR limit*, and the *internal SR limit*.

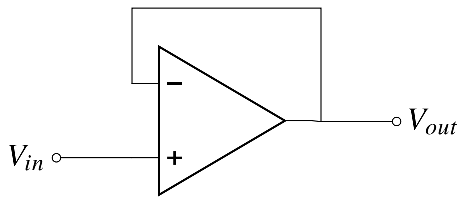
Chapter 3

Opamp stages

Opamps may be used in a large amount of different stages. This chapter shows the most common reference designs which may be useful to the reader. Keep in mind that, of course, many more configurations could be derived from these to adjust them to your individual needs.

3.1 Voltage buffer

This circuit has already been seen in chapter 2. It provides a basic 1 to 1 voltage transfer, with very high input impedance and very low output impedance, making it great as a voltage reader and ideal voltage generator:



$$G_{loop}(s) = -A(s)$$

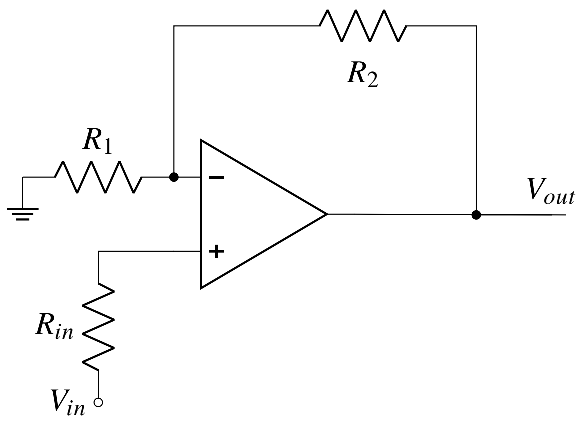
$$G_{id} = 1$$

$$BW = A_0 f_0$$

Figure 3.1: Opamp voltage buffer.

3.2 Non inverting configuration

This configuration was also seen in chapter 1. It provides a basic way to amplify a voltage signal without inverting it.



$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1+R_2}$$

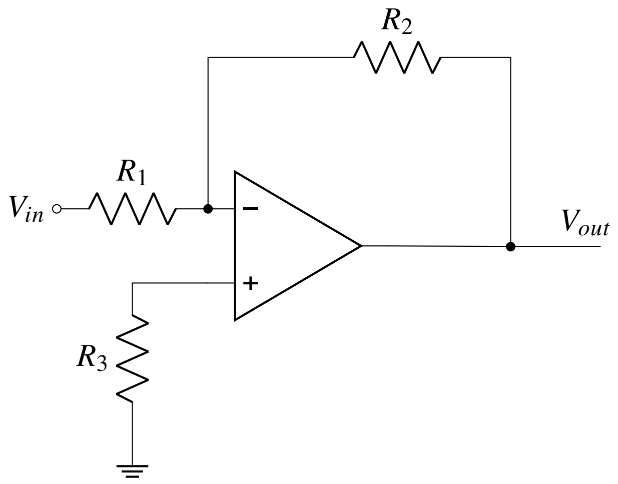
$$\mathrm{BW} = G_{loop}(0) \cdot f_0$$

$$G_{id} = 1 + R_2/R_1$$

Figure 3.2: Opamp non-inverting stage.

3.3 Inverting configuration

This configuration is similar to the non inverting configuration, but it *inverts* the output, meaning its gain is negative. R_3 has been placed to compensate the bias currents, and it should be chosen as $R_3 = R_2//R_1$.



$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1+R_2}$$

$$\mathrm{BW} = G_{loop}(0) \cdot f_0$$

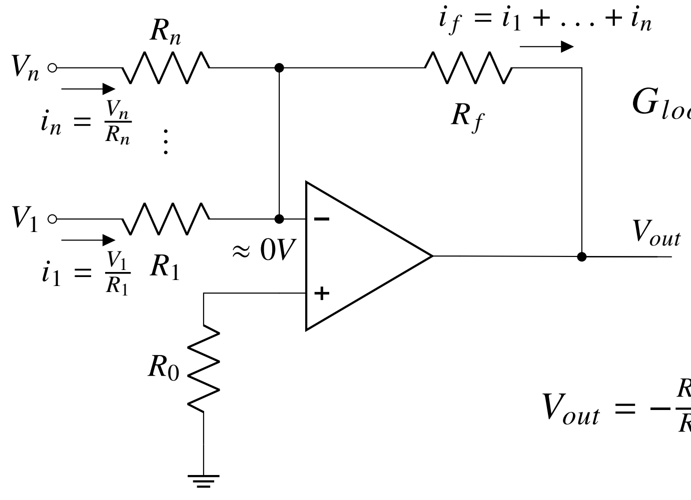
$$G_{id} = -R_2/R_1$$

Figure 3.3: Opamp inverting stage.

Note that both this and the non-inverting configurations suffer from a significant voltage offset at the output, equal to $V_{out} \pm V_{os}(1 + R_2/R_1)$, which is in the order of several volts. This hihliights the need for either AC coupling -which means putting a capacitor in series between two stages to isolate the DC bias voltage- or to use a proper offset nulling network like the one shown in Chapter 2.

3.4 Voltage adder

The voltage adder can be built in several configuration, which originate from the inverting and non-inverting stages. The configuration seen here is the inverting one, meaning the output is proportional to minus the sum of the inputs.



$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1//R_2//...//R_n}{R_1//R_2//...//R_n+R_f}$$

$$R_{in} = R_i, \; R_{out} \approx 0$$

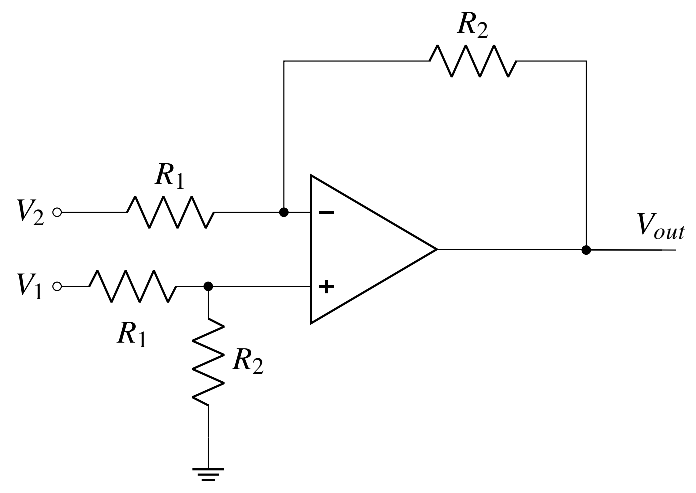
$$V_{out} = -\frac{R_f}{R_1}V_1 - \ldots - \frac{R_f}{R_n}V_n$$

Figure 3.4: Opamp inverting voltage adder.

Again, R_0 should be sized to compensate the bias currents, hence $R_0 = R_f//R_1//R_2//\ldots//R_n$. Also, note that altough the figure only shows two inputs, as many as needed could be inserted. The only detail to take care of, when putting many inputs, is not to reduce the G_{loop} too much. Thus large $R_1 \ldots R_n$ should be chosen, so that the voltage partition within the loop gain formula isn't too small.

3.5 Voltage subtractor

This configuration is also called the *difference amplifier*, as it is used to amplify the difference between two voltage signals. The idea here is to mix an inverting and a non inverting configurations. Since the gain of the non inverting stage is larger, a voltage partition should be added at the input to reduce. One disavantge of this setup is that the input resistances are not infinite for either input, and they are different.



$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1+R_2}$$

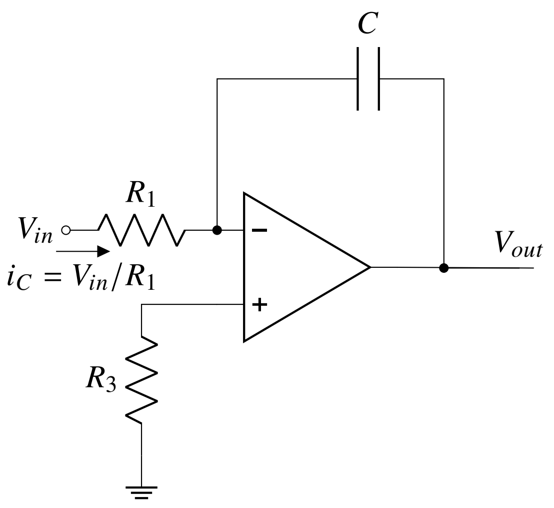
$$V_{out} = \frac{R_2}{R_1}(V_1 - V_2)$$

Figure 3.5: Opamp voltage subtractor.

The V_2 input sees an input resistance of R_1, while the V_1 input sees an input resistance of $R_1 + R_2$, which can be an issue for delicate differential sensors. To address this issue, the INA will be introduced in one of the next chapters.
There is also a common mode gain resulting from mismatched of the resistors along the two paths, which is equal to $G_{cm} = 2\frac{R_2}{R_1}\epsilon$. This result can be achieved by applying a V_{cm} signal, equal for both V_1 and V_2, while saying that the resistors have a percentage error ϵ between pairs.

3.6 Integrator

This stage is used to compute the integral of the input in time. By remembering that the equation defining the capacitor is $i_C = C \cdot \frac{dV_C}{dt}$, and by using the virtual ground of the stage, we can work out that $\frac{V_{in}}{R_1} = -C \cdot \frac{dV_C}{dt}$. This is just the current balance between R_1 and the capacitor. By integrating both sides of this equation, we can see that $V_{out}(t) = V_{out}(0) - \frac{1}{RC}\int_0^t V_{in}(t)dt$.

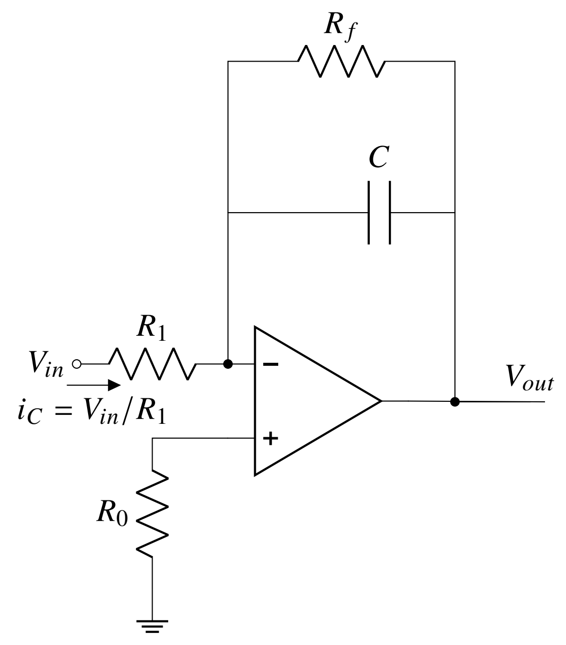


$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1+\frac{1}{sC}}$$

Figure 3.6: Opamp inverting ideal integrator stage.

In Laplace domain, we can simply consider this as a generalized version of the inverting stage, whose gain is $G_{id}(s) = -\frac{Z_C}{R_1} = -\frac{1}{sR_1C}$. In fact, in the Laplace domain, integrating is equal to dividing by s. The Bode plot of this stage is thus very simple: a straight line with slope -20dB/dec.

One issue with this design is that a DC voltage that is not zero (such as the one always present due to the V_{os} of the opamp), is integrated in time until the opamp reaches its power supply and saturates. This is sometimes undesirable, as it means this circuit can only be used for very brief amount of times before it saturates, unless some external circuit is built to restore its correct bias. One way to solve this issue is to move the pole from 0Hz to a higher frequency. Let's consider the circuit below:



$$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1 + \frac{1}{sC}//R_f}$$

Figure 3.7: Opamp inverting real integrator stage.

The gain of this stage can be computed as a generalized inverting stage:

$$G_{id}(s) = -\frac{Z_f(s)}{R_1} = -\frac{R_f//\frac{1}{sC}}{R_1} = -\frac{R_f}{R_1}\frac{1}{1+sCR_f}.$$

Note that the pole is what could be expected by inspecting the circuit: the capacitor C sees virtual ground on one side, and the output of the opamp on the other side: only R_f sists across it and takes part in the pole. Also note that there'll be a zero in the G_{loop} corresponding to this pole. This is because the poles of a closed loop system "flow" towards the zeros as the loop gain increases. Having a zero of G_{loop} at CR_f means that a pole will appear there in the real gain.
Note that at frequencies above that of the pole ($f >> \frac{1}{2\pi CR_f}$), the denominator of the ideal gain reduces to $1 + sCR_f \approx sCR_f$, thus restoring the integrating behavior. At low frequency, instead, the gain is just $-R_f/R_1$, which must be chosen to prevent the output from saturating.
Something else worth noting is that the stage behaves as a low pass filter: DC signals have a fixed gain, while signal at higher frequencies than the pole are reduced by the 1/s trend.

3.7 Derivator

The derivator can be concieved in a similar fashion as the integrator. The idea is to force a voltage on a capacitor, whose current will then depend on the derivative of the input voltage. Such current is then amplified by a resistor across the opamp:

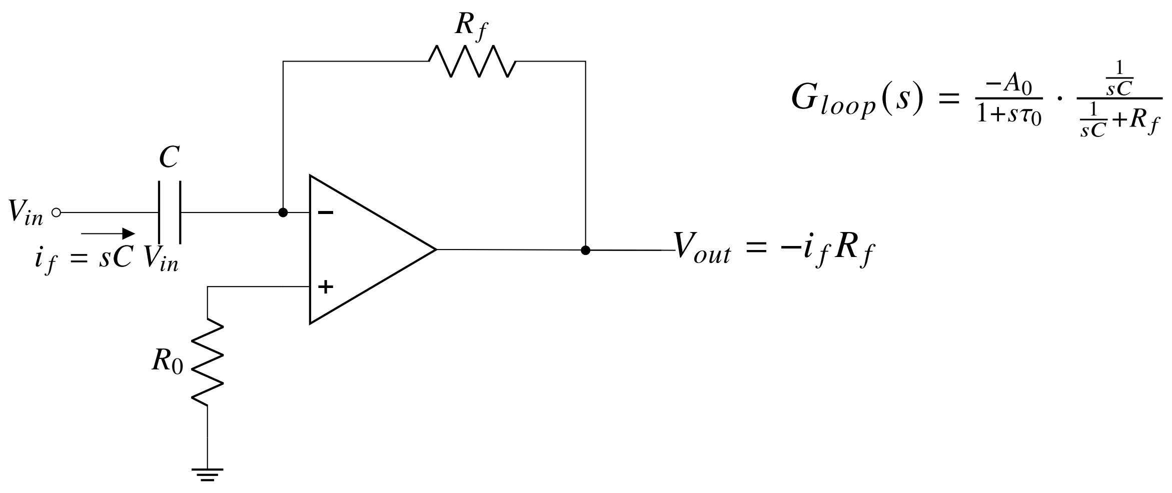

R_f
C
V_{in}
$i_f = sC\,V_{in}$
R_0
$V_{out} = -i_f R_f$
$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{\frac{1}{sC}}{\frac{1}{sC}+R_f}$

Figure 3.8: Opamp ideal derivator stage.

The ideal gain can be computed as a generalized inverting stage, so $G_{id} = -\frac{R_f}{Z_C(s)} = -sCR_f$. In time domain, the current balance at the - node of the opamp reads as $i_C = C\,\frac{dV_{in}}{dt} = i_f = -\frac{V_{out}}{R_f}$. Thus $V_{out}(t) = -RC\frac{dV_{in}}{dt}$.

One problem with this design is that signals at very high frequencies, like a voltage step, or even white thermal noise -which can extend at high frequency- generate a very large output, meaning that even a small $1\mu V$ fluctuation, if it's fast enough, can lead the opamp to saturate. A possible solution is to limit the high frequency gain by introducing a pole [1]:

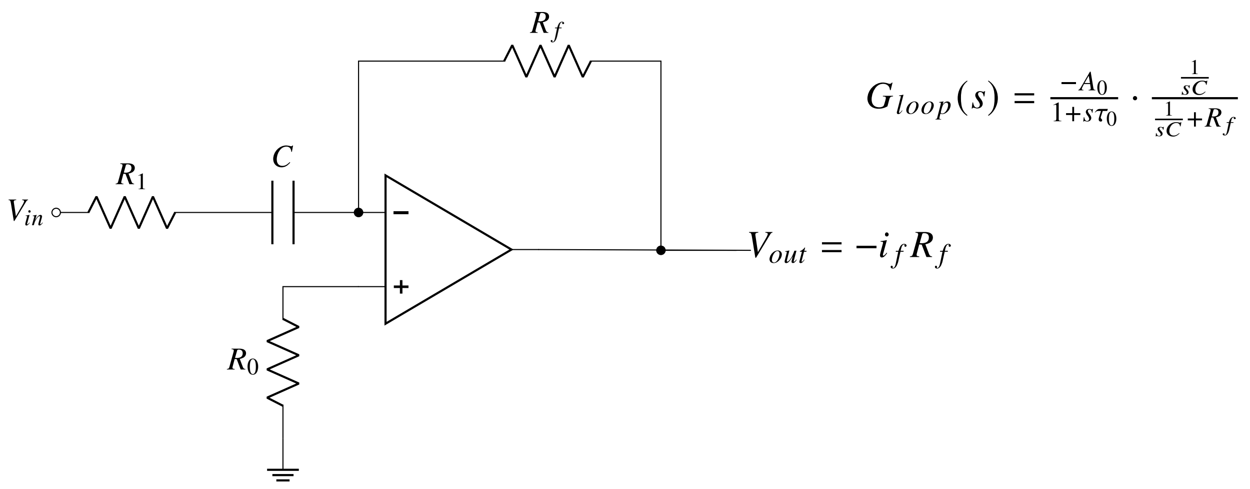

R_f
R_1
C
V_{in}
R_0
$V_{out} = -i_f R_f$
$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{\frac{1}{sC}}{\frac{1}{sC}+R_f}$

Figure 3.9: Opamp real derivator stage.

[1] note that even the ideal derivator doesn't have infinite bandwidth, eventually more poles are present in the real gain due to G_{loop} crossing the 0dB axis.

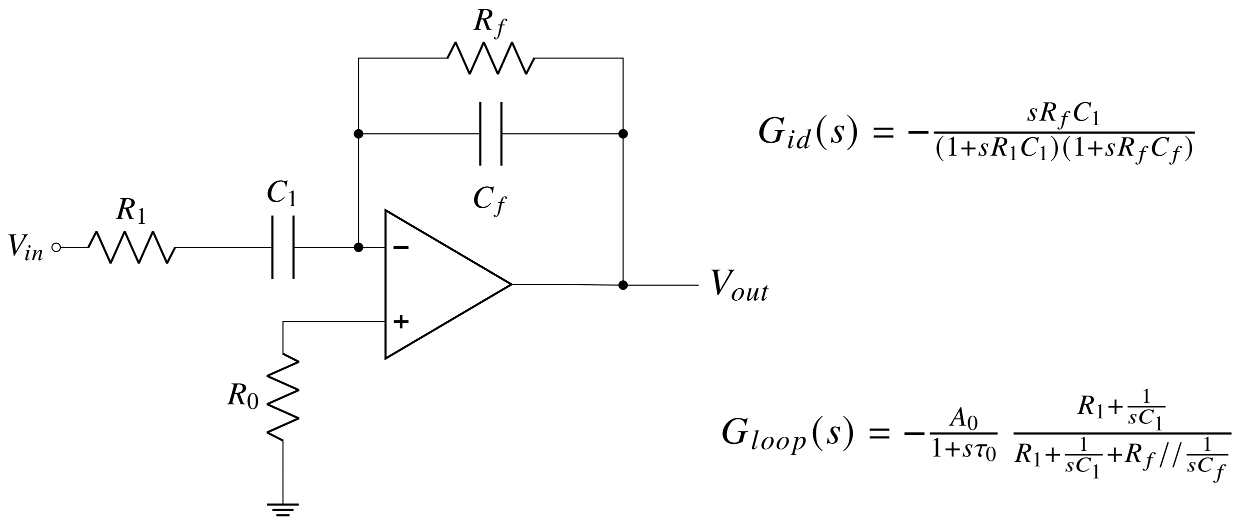
The gain can be computed by using the generalized inverting formula, as:

$$G_{id}(s) = -\frac{R_f}{Z_1(s)} = -\frac{R_f}{R_1 + \frac{1}{sC}} = -\frac{sCR_f}{1 + sCR_1}$$

Showing a derivator-like behavior for frequencies lower then the pole[2], and a constant gain of $-\frac{R_f}{R_1}$ at higher frequencies. Note that this sort of behavior is consistent with a so-called *"high pass filter"*, and this stage can be used for that too.

3.8 Band-pass filter

The band pass filter is based on the idea of combining both the 1/s and s behaviors of the integrator and derivator to make a stage that can amplify signals only between two frequencies:



R_f
C_1
C_f
R_1
V_{in}
V_{out}
R_0
$G_{id}(s) = -\frac{sR_fC_1}{(1+sR_1C_1)(1+sR_fC_f)}$
$G_{loop}(s) = -\frac{A_0}{1+s\tau_0}\frac{R_1+\frac{1}{sC_1}}{R_1+\frac{1}{sC_1}+R_f//\frac{1}{sC_f}}$

Figure 3.10: Active bandpass filter.

The ideal gain can be computed as a generalized inverting stage:

$$G_{id} = -\frac{Z_f(s)}{Z_1(s)} = -\frac{R_f//\frac{1}{sC_f}}{R_1 + \frac{1}{sC_1}} = \ldots = -\frac{sR_fC_1}{(1 + sR_1C_1)(1 + sR_fC_f)}$$

This reveals a zero in the origin, and two poles, introduces by R_1 and R_f. If the pole due to R_1C_1 is the first one to activate, the mid frequency gain will be $-\frac{R_f}{R_1}$. At lower frequencies, we'll have an increasing gain due to the zero at 0Hz, and at frequencies where both poles are active the behavior will be 1/s. Note that if the pole due to R_fC_f is the first one to become active, the mid frequency gain will be instead $-\frac{R_fC_1}{R_fC_2} = -\frac{C_1}{C_f}$. This is usually below

[2]at frequencies below the pole $1 + sCR_1 \approx 1$

the 0dB axis and is thus undesirable. The stage **must** be sized to ensure that the first pole to become active is that due to C_1.

With $R_1 = 1.5\ k\Omega, R_f = 15\ k\Omega, C_f = 3.3\ nF, C_1 = 100\ nF$, the poles are expected to be at 3.215 kHz and 1.061 kHz. The mid frequency gain is expected to be -10, which is 20dB. The bode plot is the following:

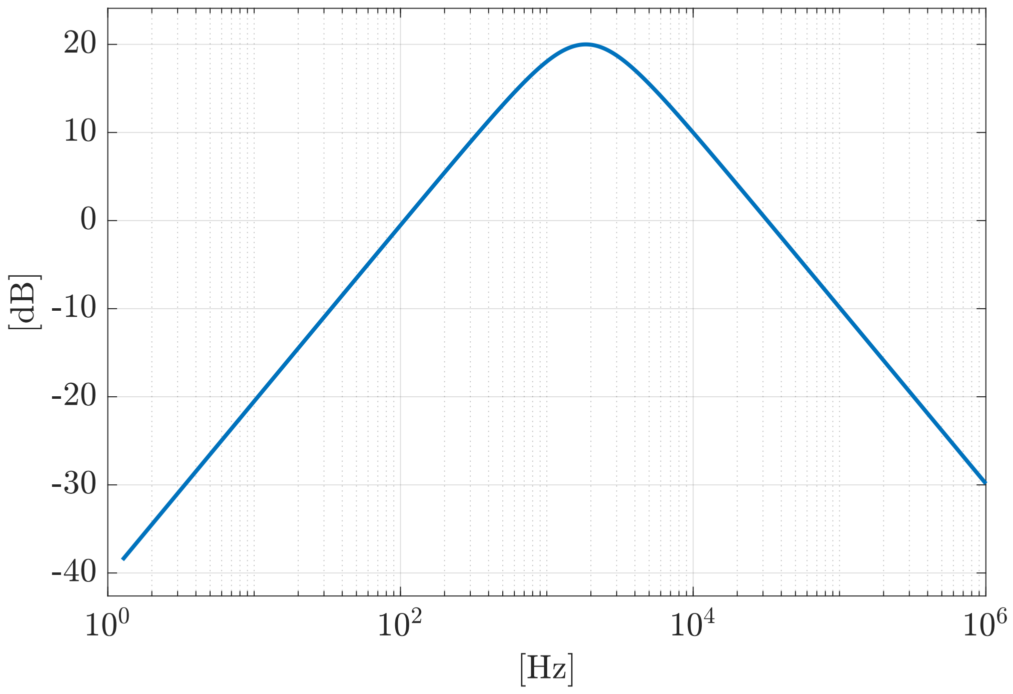


Figure 3.11: Bode plot of the band pass filter.

Note that the two poles, being close in frequency, don't quite activate in time, leading to an actual transfer at mid frequency that is not constant -i.e. flat- over the pass region. Also note the ±20dB/dec slopes at the sides, meaning that a signal well above the expected range is still amplified. From this plot, we can see that the stage amplifies all signals from $\approx 10^2\ Hz$ to $\approx 3 \cdot 10^4\ Hz$. To see this, we can just look at the 0dB crossings.

3.9 Current buffer

This stage uses the opamp's virtual ground to collect a current and turn it into a voltage signal at the output by means of a resistor. It is used any time a small signal current, such as that of an optical diode sensor, needs to be collected and amplified/filtered by further stages. Note that the input impedance of this stage is zero, which allows it to read a current well without any loss due to a possible non ideal current source (which would have a resistor in paralel). The output resistance is, instead, zero, due to the feedback constraining the voltage at the output node.

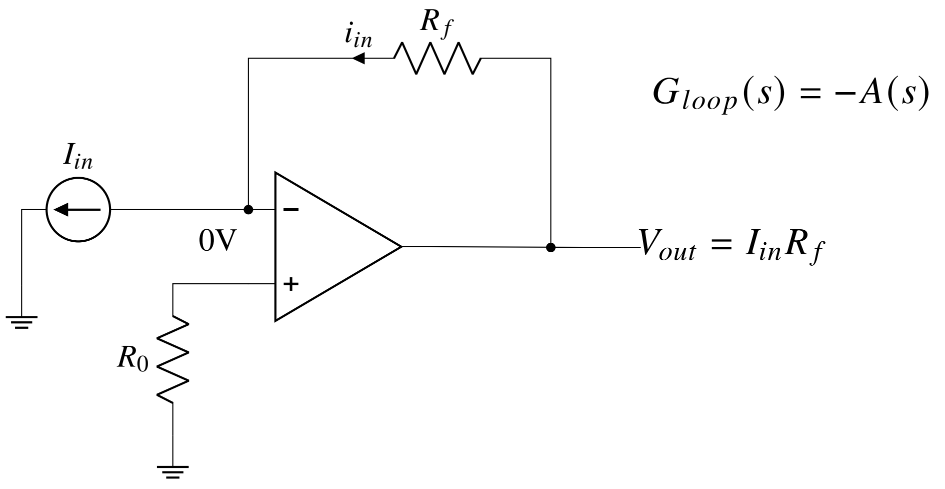


Figure 3.12: Opamp current-to-voltage converter (Current Buffer).

3.10 Current generator

This stage takes an input voltage and uses it to generate a current, which is then forced into a load R_L. Note the strange position of the load.
It is used in audio applications, where certain loads must be controlled by forcing a current, rather than a voltage. Note that the load sees a current which does not depend on its voltage, meaning the resistance seen is infinite. As far as R_L is concerned, this stage acs like an ideal current generator.

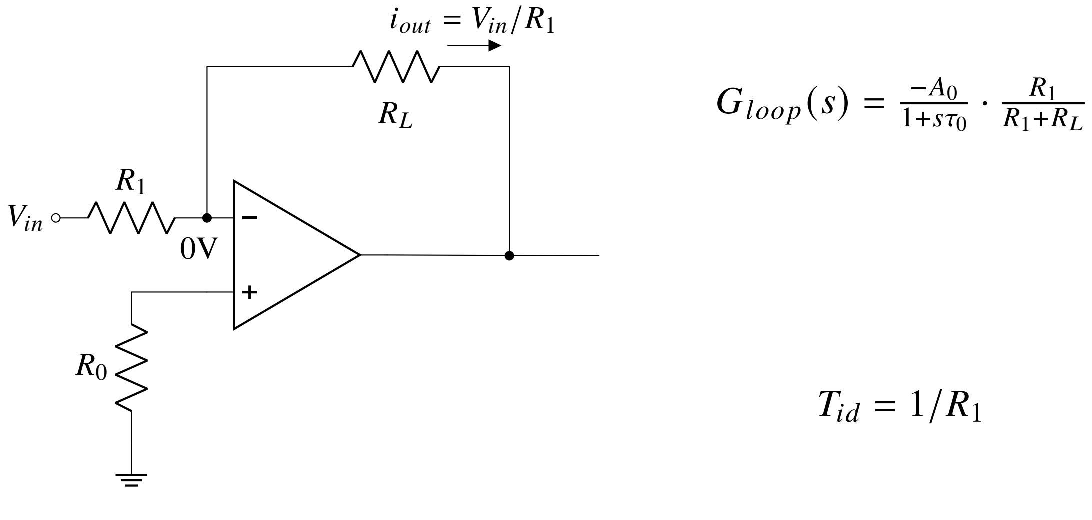
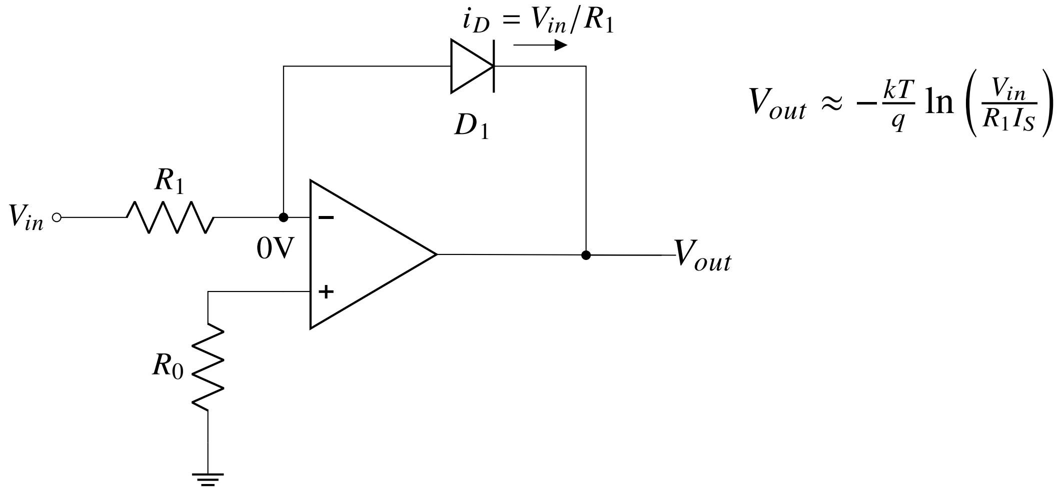

$i_{out} = V_{in}/R_1$
R_L
$G_{loop}(s) = \frac{-A_0}{1+s\tau_0} \cdot \frac{R_1}{R_1+R_L}$
R_1
V_{in}
0V
R_0
$T_{id} = 1/R_1$

Figure 3.13: Opamp voltage-controlled current source.

3.11 Log amplifier

This stage forces a current into a load with an exponential I-V curve, like a diode or a BJT transistor, to obtain a logaritmic voltage signal. It can be used to calculate logarithms in analog electronics! Remember: a diode has an I-V curve like $i_D(V_D) = I_S(e^{\frac{qV_D}{kT}} - 1)$, where the -1 is negligeble for a positive bias.



$i_D = V_{in}/R_1$
D_1
R_1
V_{in}
0V
R_0
V_{out}
$V_{out} \approx -\frac{kT}{q} \ln\left(\frac{V_{in}}{R_1 I_S}\right)$

Figure 3.14: Opamp logarithmic amplifier stage.

3.12 Exp amplifier

This stage forces a voltage into a load with an exponential I-V curve, like a diode or a BJT transistor, to obtain an exponential current signal, which si then amplified by a resistor into a voltage signal. It can be used to calculate exponentials in analog electronics! Remember: a diode has an I-V curve like $i_D(V_D) = I_S(e^{\frac{qV_D}{kT}} - 1)$, where the -1 is negligeble for a positive bias. In this stage, $V_D \approx V_{in}$ due to the virtual ground.

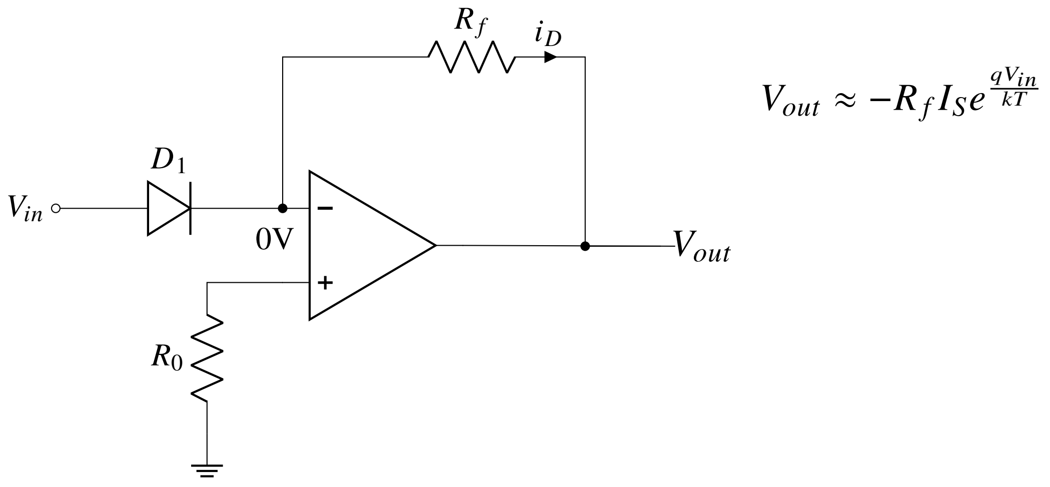


Figure 3.15: Opamp exponential amplifier stage.

3.13 Sqrt amplifier

This stage uses the quadratic I-V relation of a MOS transistor. It forces a current into the transistor, generating a voltage signal proportional to the square root of the forced current.

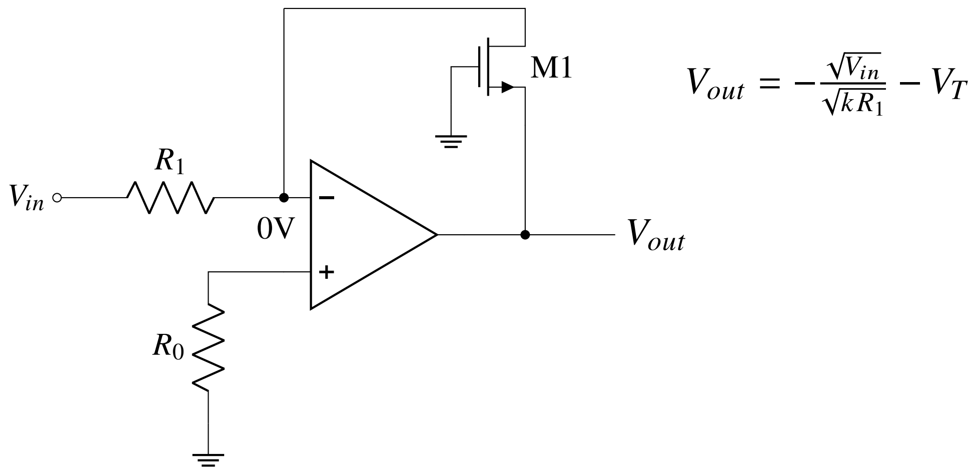


Figure 3.16: Opamp square root amplifier using an NMOS transistor.

As mentioned above, this circuit forces a current V_{in}/R_1 into the MOS, whose V_{gs} is equal to $-V_{out}$. As long the the MOS is working in saturation, this results in $V_{in}/R_1 = k(-V_{out}-V_T)^2$, hence $G_{id} = -\frac{\sqrt{V_{in}}}{\sqrt{kR_1}} - V_T$.

3.14 POW2 amplifier

This stage uses the quadratic I-V relation of a MOS transistor. It forces a voltage into the transistor, generating a current signal proportional to the square of the input voltage, which is then converted into a voltage signal by a resistor along the feedback path.

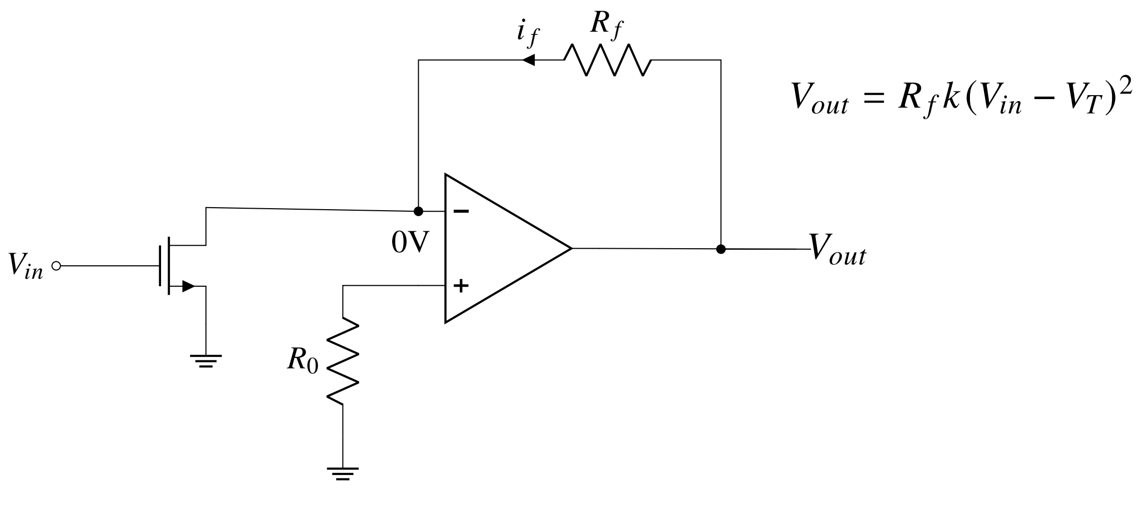


Figure 3.17: Opamp square amplifier stage using an NMOS transistor.

This circuit forces a voltage $V_{in} = V_{GS}$, which activates a current that then flows into R_f. The current balance reads $i_f = k(V_{in} - V_T)^2 = \frac{V_{out}}{R_f}$.

3.15 Single wave rectifier

This circuit uses the non-linear propreties of diodes to obtain a static transfer function of the kind

$$V_{out} = \begin{cases} V_{in} & V_{in} > 0 \\ \approx 0 & V_{in} < 0 \end{cases}$$

Note that to analyze these rectifying circuit, we can use the following simplified model of a diode:

$$\begin{cases} I_D = 0 & V_D < 0.7\ V \\ V_D \approx 0.7\ V & I_D > 0 \end{cases}$$

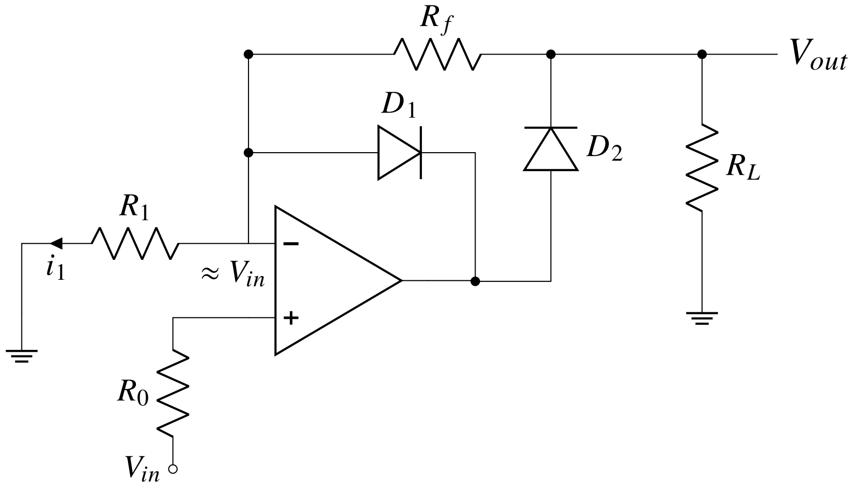


Figure 3.18: Precision half-wave rectifier (non-saturating non-inverting type).

This is achieved by having at least one of the diodes always active, thus preserving negative feedback and virtual ground, and preventing the opamp from saturating. Other designs with just one diode exist, but they don't prevent the opamp from saturating. Depending on the sign of V_{in}, which activates a current in R_1 equal to $i_1 = \frac{V_{in}}{R}$, one of the diodes is always active, while the other inactive. If $V_{in} > 0$, the current will flow through D2 and D1 will turn off. Therefore, $V_{out} = V_{in}(1 + R_f/R_1)$. When $V_{in} < 0$, D1 turns on and D2 off, and the output voltage is given by the partition between the resistors: $V_{out} = V_{in}\frac{R_L}{R_f+R_L}$. This isn't quite zero, although R_L could be sized to make this very low. There is an inverting configuration which helps with this problem:

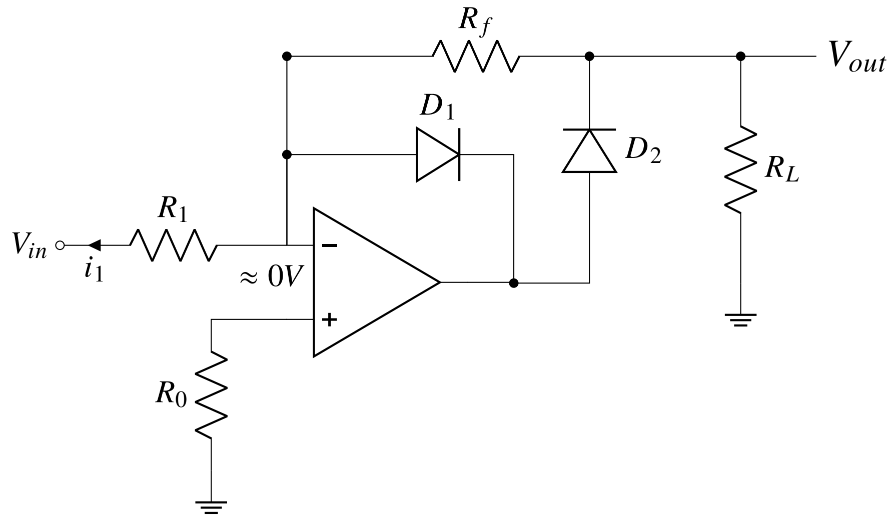


Figure 3.19: Precision half-wave rectifier (non-saturating inverting type).

Now $i_1 = -V_{in}/R_1$ and when $V_{in} < 0$, D2 turns on and D1 turns off, making the output $V_{out} = -R_f/R_1$. When $V_{in} > 0$, instead, D1 will turn on and D2 off. Since the voltage at the minus node of the opamp is fixed to zero, no current can flow into R_f and R_L, making $V_{out} = 0$. This is called *inverting* rectifier for obvious reasons.
Note that no condition when both diodes are active -or inactive- can exist. To demonstrate that, simply try to solve the circuit in either state, and you'll find that the currents found don't match the direction of the diodes, or that the polarities of the voltages found don't agree with the ideal diode model.

3.16 Double wave rectifier

This is a more complex diode circuit, used to calculate the abs(V_{in}), instead of just doing so for only $V_{in} > 0$ as the circuit in figure 3.18, or only for $V_{in} < 0$, like the circuit in 3.19 does. There are several designs to achieve this, which require at least two opamps. One approach is to add up, in some way, currents and voltages produces by those two simple stages, but there are other approaches. One design is the following:

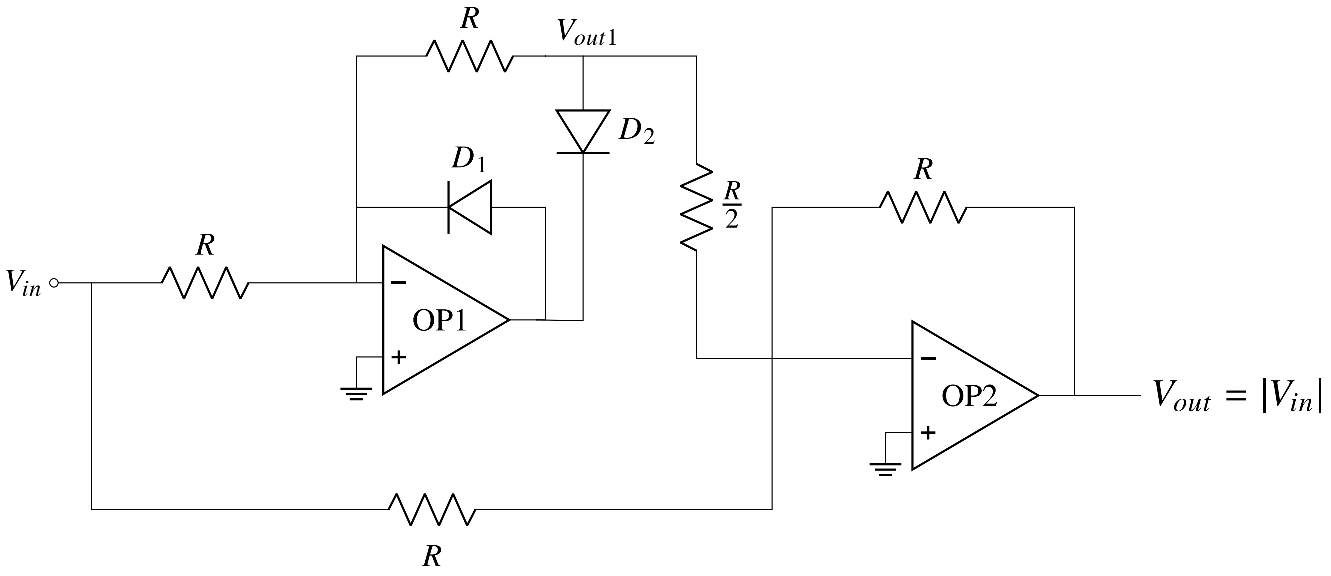


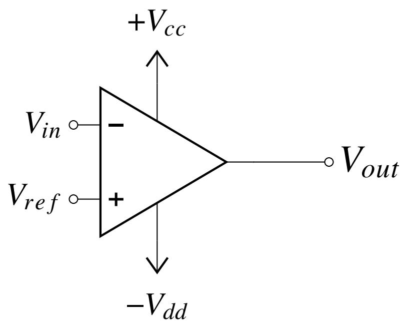
Figure 3.20: Precision Full-Wave Rectifier (Double Rectifier) stage.

The idea of this circuit is to use the second stage, which is an adder, to add up the input signal and the output of the single wave rectifier on top, which is at node V_{out1}, weighted by 2, with the input signal itself. When $V_{in} < 0$, D1 turns on while D2 turns off, resulting in no current through the R and $R/2$ in the feedback of OP1. This also means that the output is only driven by the bottom adder branch, resulting in $V_{out} = -V_{in}$. When $V_{in} > 0$, D1 turns off and D2 turns on. This means that $V_{out1} = -V_{in}$, as all the current V_{in}/R is forced into the feedback path of OP1 containing R and D2. This signal is amplified by the inverting adder along with the signal from the bottom path, resulting in $V_{out} = -V_{in} - 2V_{out1} = -V_{in} + 2V_{in} = V_{in}$. Hence, the circuit computes the absolute value of the input voltage. Note that to achieve this, the diodes must be reversed compare to the single wave rectifier of figure 3.19, as V_{out1} is required to be negative, not positive, due to the inverting second stage.

3.17 Comparator

This circuit uses the opamp in open loop to saturate at $\pm V_{sat}$ when the input is over -or below- a certain threshold, which can be set at the other input of the opamp[3]. The value of V_{sat} depends on the opamp being used, some opamps, called *Rail to Rail* saturate at their power supply, but most designs don't. Also note that due to the finite gain of the opamp, and due to offset, the threshold won't be perfect. In particular, the offset will change the value of the threshold by a few mV, and the finite gain will require the input to exceed the threshold by a few μV to ensure saturation.

[3]For instance, with a voltage partition, or a Zener diode



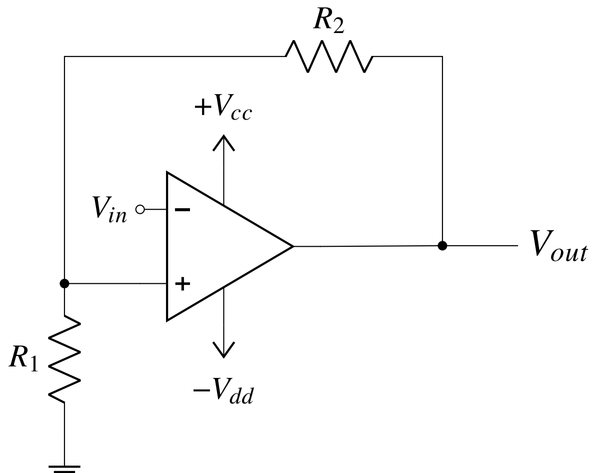
$$V_{out} = \begin{cases} +V_{sat} & V_{in} < V_{ref} \\ -V_{sat} & V_{in} > V_{ref} \end{cases}$$

Figure 3.21: Op-amp operating as an open-loop comparator.

This configuration is often used in analog to digital converters, as it provides an easy way to convert an input analog signal to a digital signal. Of course, to obtain better precision than 1 bit more comparators are needed. See the chapter on ADC for more details.

3.18 Smith trigger

This circuit uses positive feedback to always saturate at $\pm V_{sat}$ (if the opamp is rail to rail). The feedback path ensures that the switching thresholds change depening on the current status of the output, thus resulting in a behavior that depends on the previous history of the circuit: Hysteresis.



Thresholds:

$$V_{TH} = +V_{sat} \frac{R_1}{R_1+R_2}$$

$$V_{TL} = -V_{sat} \frac{R_1}{R_1+R_2}$$

Figure 3.22: Inverting Schmitt Trigger.

When V_{in} is very large, the output will always saturate to $-Vsat$, being at the minus input. This will result in the output staying there until $V_{in} = V_{TL}$. At this point, the output swings to $+V_{sat}$, and it will stay there until the input has reached the new threshold of the system, V_{TH}. This results in the following hysteresis plot:

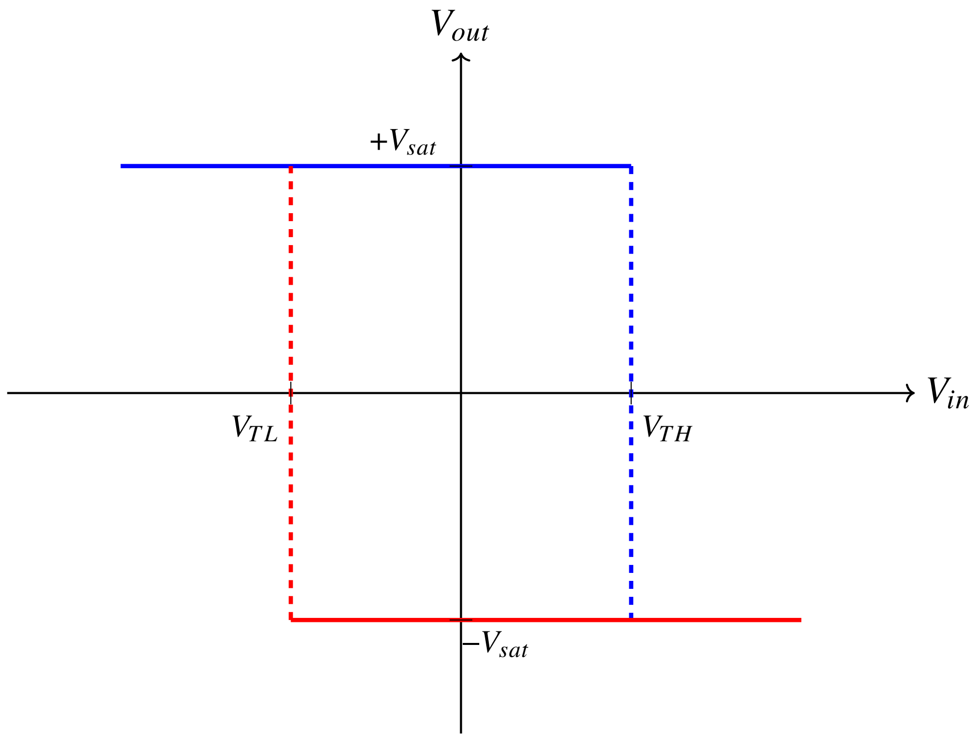


Figure 3.23: Inverting Schmitt Trigger transfer characteristic.

Since very positive V_{in} lead to negative V_{out}, and vice versa, this configuration has the name *inverting Smith trigger*, and it is widely use as a crude way to convert certain analog signals into digital ones. Note that the dottet regions of the transfer curve are only vertical if the opamp has infinite gain. Else, they will show a finite slope, and the output might get stuck there for certain inputs very close to the treshold. Still it's not a real problem in physical circuits with noise, offset, ecc...

Chapter 4

Frequency compensation

4.1 Bode stability and phase margin

A circuit with feedback has different poles than the ones found in $G_{loop}(s)$. As we have already seen in chapter 1, the closed-loop poles are found by solving equation 2.5, and they are stable if $Re[s] < 0$.
If we want to study the stability of a system without solving this equation every time, which may be inconvenient, we can use the Bode criterion, which states that: *"The closed loop system will be stable if the phase of $G_{loop}(f)$, at the frequency where $|G_{loop}(f)|$=0dB, is greater than $-360°$."* All of this, of course, substituting $s = 2\pi jf$ into the loop gain formula.
To quantify how far the system is to instability, we define the phase margin as the difference between the phase of $G_{loop}(f)$ at the 0dB crossover frequency and -360 deg:

$$\theta_M = \angle G_{loop}(f_{0dB}) + 360deg \tag{4.1}$$

Keep in mind that the phase of G_{loop} on a Bode plot will be negative, and, without any poles or zeros, should be $-180°$, as the feedback in DC is usually negative

4.2 Root locus

Another way to study the stability of a closed loop system is the root locus. It plots the solutions -on the complex plane- of $\alpha G_{loop}(s) = 1$ for growing values of α, allowing us to see exactly how the closed-loop poles change as the circuit's feedback grows stronger. Of course the actual poles are those found for $\alpha = 1$. There are many rules to plot the root locus manually, but here I'll just write the following ones:

- The number of closed-loop poles is equal to the number of open loop poles.

- The open loop poles start at the open loop poles, forming as many branches as there are poles.
- The locus is symmetric about real axis.
- The branches lay on the real axis only where there's an odd number of poles/zeros to the right.
- The branches may go towards the zeros of G_{loop} or at infinity
- The branches that go towards infinity must divide the plane in angles of $\frac{360°}{\#\text{poles} - \#\text{zeros}}$

As an example, let's consider $G_{loop}(s) = -\frac{1+s\ 3}{(1+s\ 4)(1+s\ 8)}$. The root locus is shown in figure 4.1, which shows the branches first converging, then leaving the real axis (and thus becoming complex conjugate poles), and then going back into the real axis, to go towards the zero and $-\infty$.

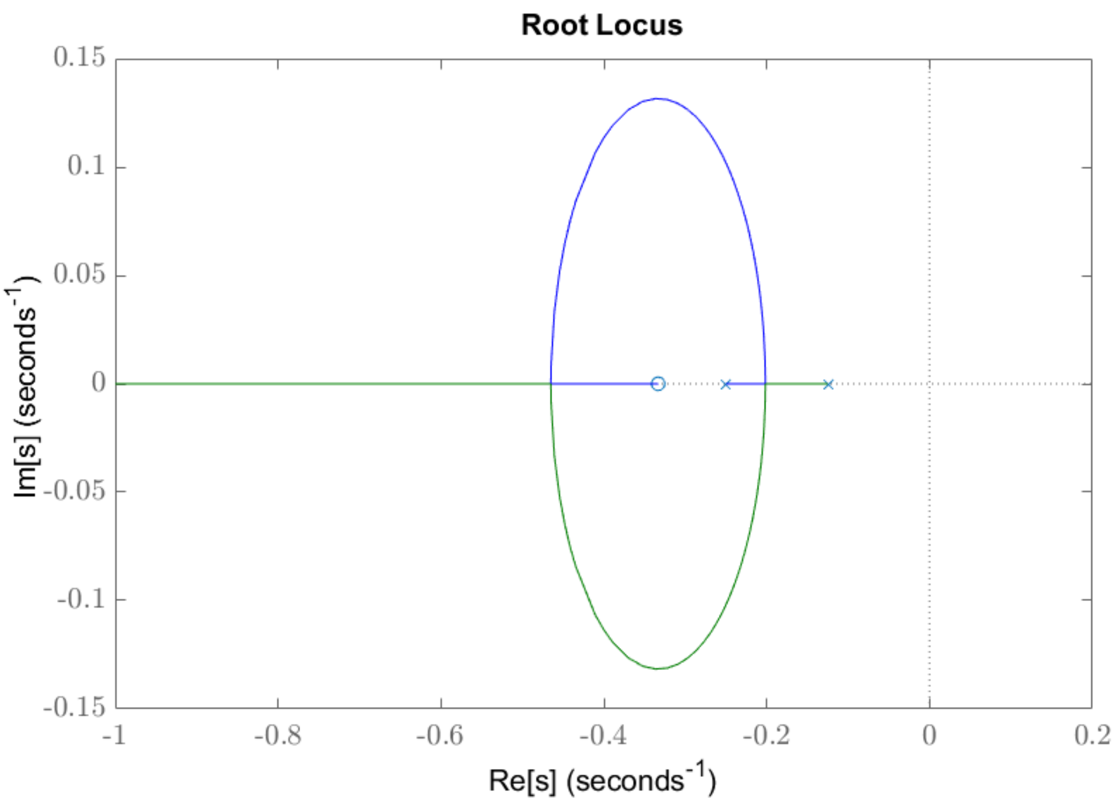


Figure 4.1: Example root locus.

4.3 A(s), β(s), closure angles

Something professor Zappa is very fond of is subdividing $G_{loop}(s) = A(s)\beta(s)$, with $A(s)$ and $\beta(s)$ chosen in a meaningful way within the circuit. This allows him to use the Bode

plot of $A(s)$ and $1/\beta(s)$, and to identify the 0dB cutoff frequency of G_{loop} as the frequency for which $|A(f)| = |1/\beta(s)|$. Since a pole introduces a -20dB/dec slope and $-90°$ phase shift into the plot, he then says that the limiting condition for stability[1] corresponds to a $G_{loop}(f)$ crossing the 0dB axis at -40dB/dec. This, in turns, corresponds to a relative slope between $A(f)$ and $\beta(f)$ of 40dB/dec, which he calls *"closure angle"*.
Please note that this is a simplified picture as it doesn't take into account singularities after the crossing, which may, of course, change this picture and make the system unstable. Overall, computing the phase margin, or even solving 2.5 guarantees a more reliable stability assessment.

4.4 Uncompensated opamps

Some opamps are *uncompensated*, meaning their response is of the kind:
$A(s) = \frac{A_0}{(1+s\tau_0)(1+s\tau_1)}$, showing two poles before the GBWP[2] at $f_0 = \frac{1}{2\pi\tau_0}$ and $f_1 = \frac{1}{2\pi\tau_1}$.
This results in a phase margin, when used in a buffer configuration like that shown in 3.1, and a Bode plot of the loop gain showing a -40dB/dec crossing of the 0dB axis: $[A_0 = 10^5,\ f_0 = 100\text{ Hz}, f_1 = 1\text{ MHz}]$

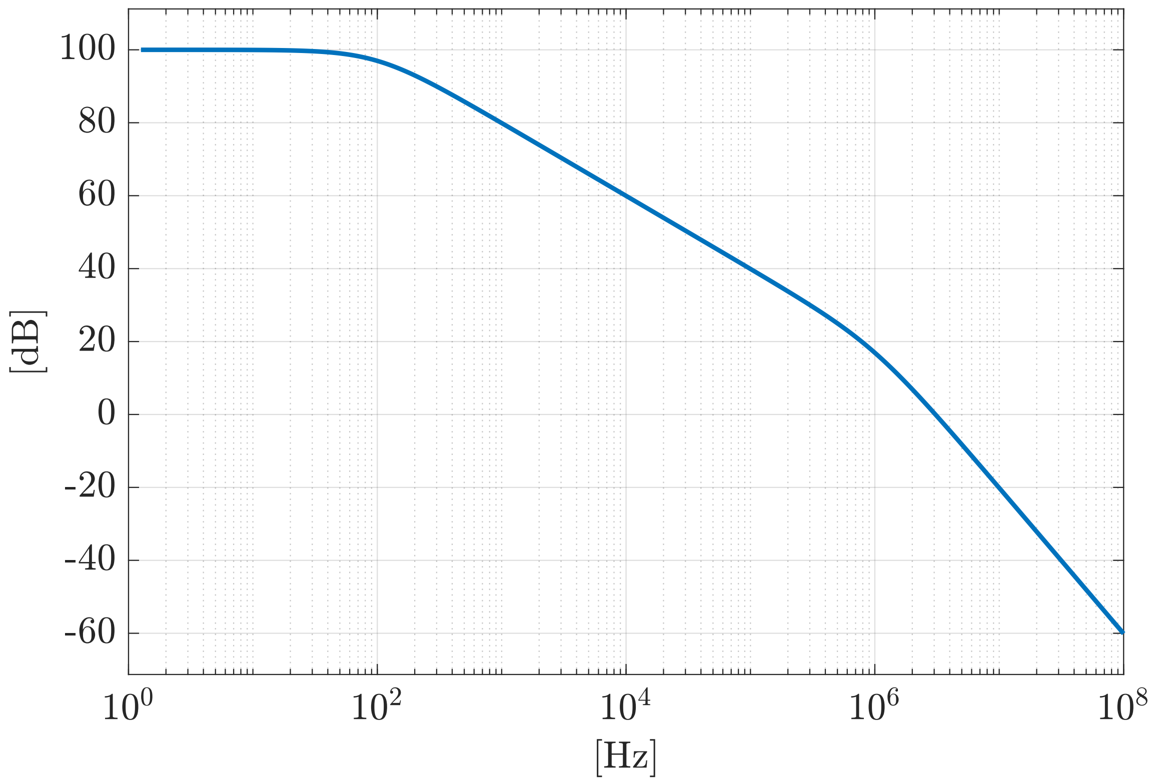


Figure 4.2: Bode plot of $|G_{loop}(f)|$ an uncompensated opamp.

[1]to have crossed and fully activated two poles, which gives $\theta_M = 0°$
[2]which would be the 0dB crossover frequency without the second pole

Note that sometimes this second pole isn't given as a frequency, but as the value of $|A(f = f_2)| = A_{min}$, which allows to compute the frequency and time constant by simply solving $A_0 \cdot f_0 = A_{min} \cdot f_1$, which can be derived from the expression for the $|G_{loop}(f)|$.

4.5 Parasitic input capacitance

It is common for opamps to show a parasitic input capacitance from the solder joint at the minus terminal and ground. This is due to the sandwich structure of the circuit boards, which often have a ground plane underneath the signal plane, acting as a capacitor to ground.[3] We can model this as follows:

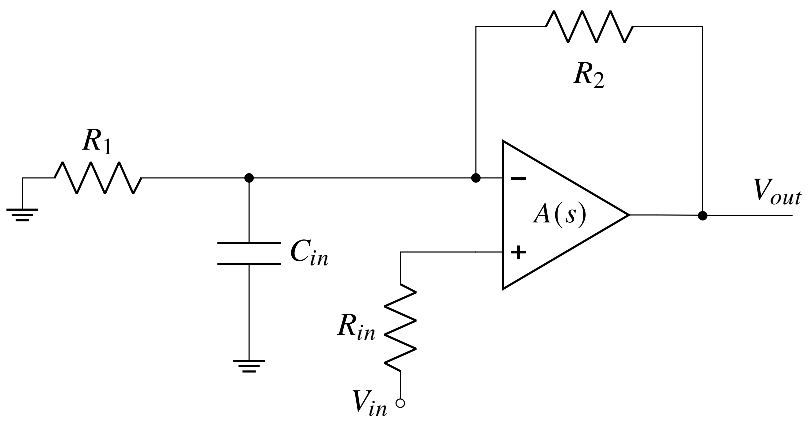


Figure 4.3: Opamp non-inverting stage with a parasitic capacitance at the minus terminal.

Let's assume that this opamp is uncompensated, meaning its $A(s)$ shows two poles as discussed above. The loop gain can be computed as a voltage partition:

$$G_{loop} = -A(s) \cdot \frac{R_1 // \frac{1}{sC_{in}}}{R_1 // \frac{1}{sC_{in}} + R_2} = \ldots = -\frac{A_0}{(1 + s\tau_0)(1 + s\tau_1)} \cdot \frac{1}{1 + sC_{in}(R_1 // R_2)} \cdot \frac{R_1}{R_1 + R_2}$$

Where the poles and zeros of C_{in} can be computed by inspection, rather than with all the computations. The pole is what could be expected at open loop, and the zero is at infinite frequency, which we can tell by shorting C_{in}. This is the bode plot: $[A_0 = 10^5,\ f_0 = 100\,\text{Hz}, f_1 = 1\,\text{MHz}, C_{in} = 3pF, R_{in} = 1\,k\Omega, R_1 = 200\,k\Omega, R_2 = 1\,M\Omega]$

[3]Note that a parasitic capacitance at the + terminal plays no role in the loop gain.

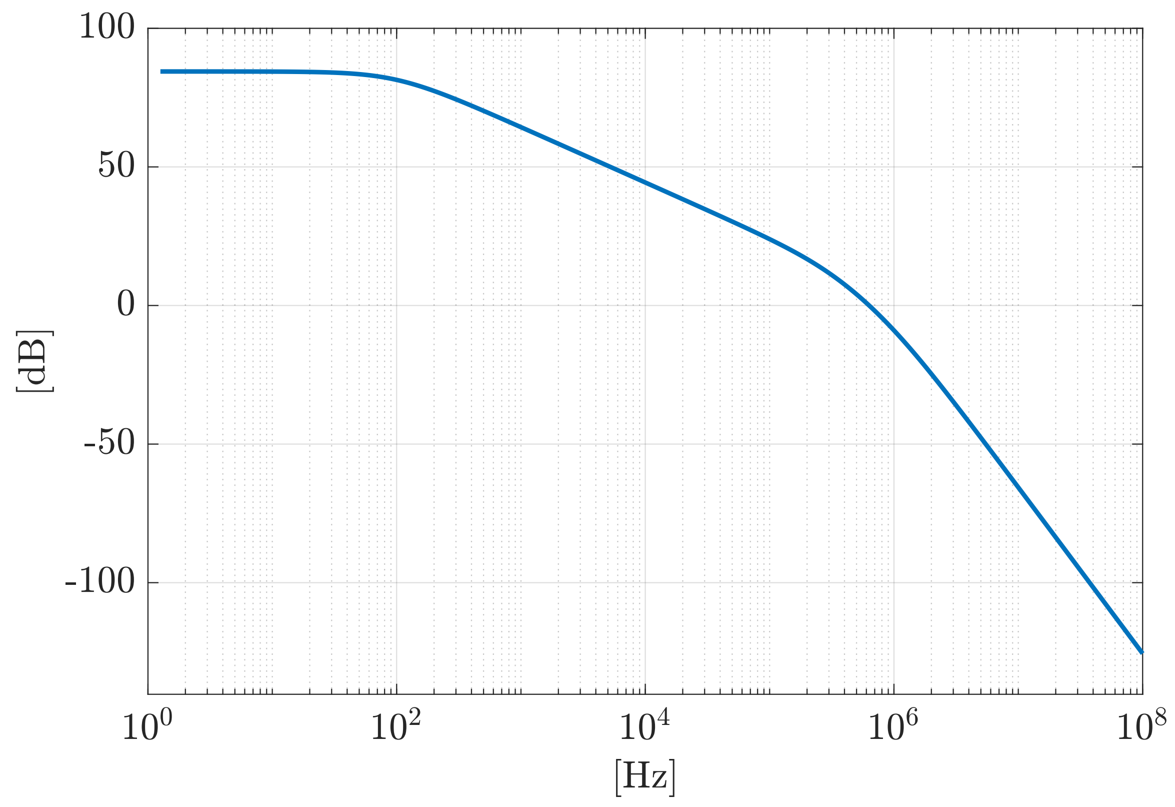


Figure 4.4: Bode plot with a stray capacitance at the input terminal.

This plot has a pole at 100 Hz, another pole at $\approx$ 320 kHz and another pole at 1 MHz, for a -60dB/dec slope and an insufficient phase margin. This circuit is unstable.

4.6 Compensation technique: C_f

To make a circuit such as the one is figure 4.3 stable some *compensation techniques* should be employed to modify the Bode plot of figure 4.4 into something more stable. One possibility is to just change the value of R_1, as that can easily reduce $G_{loop(0)}$ to achieve a 0dB crossover point before the stray poles are activated. This approach changes the ideal gain at low frequency, of course, and may therefore not be a good option. Another approach is shown below, where a capacitor C_f is added in the feedback path to cancel out the pole due to C_{in}.

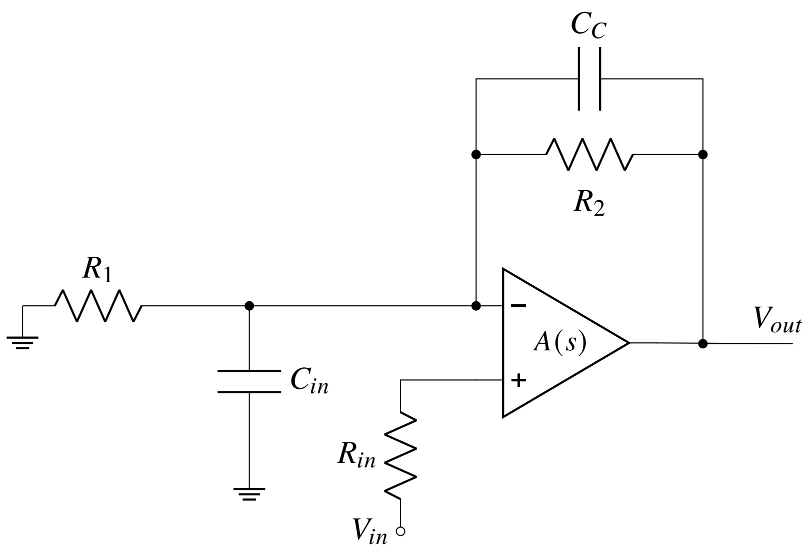


Figure 4.5: Compensation with feedback capacitor.

Note that C_C will introduce a zero in the loop gain at $\tau_Z = C_C R_2$ and a pole, being in parallel with C_{in}, at $\tau_P = (C_C + C_{in})(R_2//R_1)$. We can size C_C so that these singularities cancel each others out, obtaining, from $\tau_Z = \tau_P$, that $(C_C + C_{in})(R_2//R_1) = C_C R_2$, and therefore

$$C_C = C_{in}\frac{R_1}{R_2} \tag{4.2}$$

With this exact value of C_C, C_{in} and C_C don't add any singularities to the loop gain, and the circuit is stabilized, as the only singularities left in $G_{loop}(s)$ are those due to the opamp's $A(s)$. Note that this cancellation, in practice, won't be perfect, leaving the system with both a zero and a pole very close to each others. The presence of the zero, in particular, will lead to a different closed-loop bandwidth than before. In fact, the new ideal gain of the stage, which can be computed as a generalized non inverting stage, is:

$$G_{id} = 1 + \frac{R_2//\frac{1}{sC_C}}{R_1//\frac{1}{sC_{in}}} = \ldots = (1 + \frac{R_2}{R_1})\frac{1 + s(C_C + C_{in})(R_1//R_2)}{1 + sC_C R_2}$$

Note that C_{in} doesn't introduce a pole due to the opamp forcing its voltage directly, but it does play a role in the zero. If we substitute in equation 4.2, this reduces to:

$$G_{id} = 1 + R_2/R_1$$

This is because the poles and zeros due to those capacitors are the same, reversed, as those found in the loop gain. Having sized C_C so that they cancel out, we don't find them in the ideal gain either[4].

4.7 Compensation technique: RC and C

Another possibility to compensate an opamp is shown below. It consists of a RC network between the opamp's input.

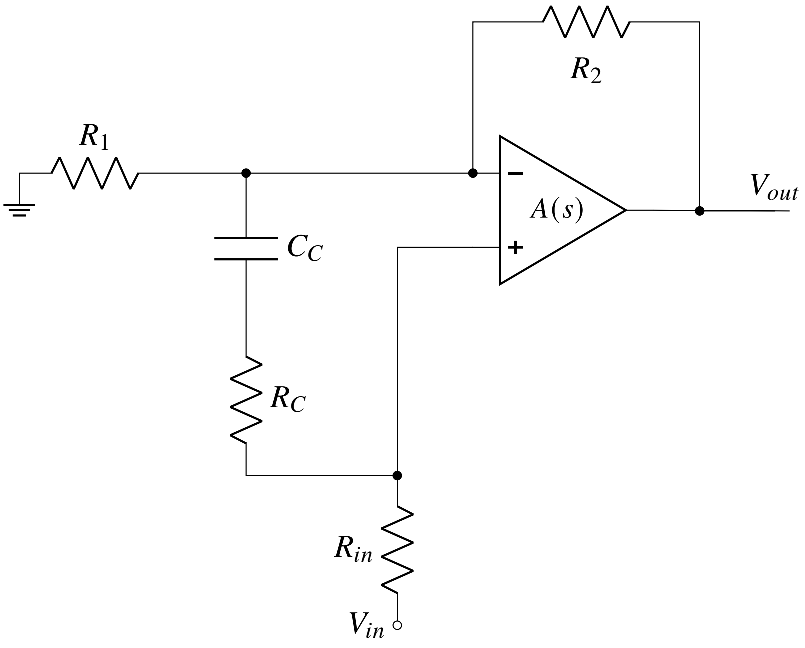


Figure 4.6: RC compensation.

Note that C_C and R_C are, ideally, always shorted by the opamp, meaning they never contribute to the ideal gain. The loop gain, instead, is modified. In DC, it is $G_{loop}(0) = -A_0\frac{R_1}{R_1+R_2}$. C_C introduces a pole in the loop gain at $\tau_P = C_C(R_C + R_1//R_2)$, and a zero, when the impedance $R_C + \frac{1}{sC_C}$ is null, for $\tau_Z = C_C R_C$. Therefore:

$$G_{loop}(s) = -\frac{A_0}{(1+s\tau_0)(1+s\tau_1)}\frac{1+sC_C R_C}{1+sC_C(R_C+R_1//R_2)}$$

This pole and zero should be placed before the 0dB transfer, to ensure the absolute value of the loop gain is reduced by the pole (which has a larger τ and therefore lower frequency), while the zero is activated before the crossing to recover the phase margin. Note that since the compensation network is shorted in the ideal transfer, it is not expected to play a significant role in the real transfer of the stage either. This is the result of a pole-zero cancellation when applying equation 2.1 to compute the real gain of the stage.

Since the idea of this stage is to reduce the loop gain to make the 0dB crossing before the

[4]assuming ideal cancellation, of course

second pole of $A(s)$, one might simply compensate by just putting a resistor between the + and - terminals of the opamp. This is what I call *R-compensation*, and it just reduces the value of $G_{loop}(0)$ by introducing a voltage partition of the kind $\frac{R_C//R_1}{R_C//R_1+R_2}$. The circuit is just like the one in figure 4.6, but with C_C shorted:

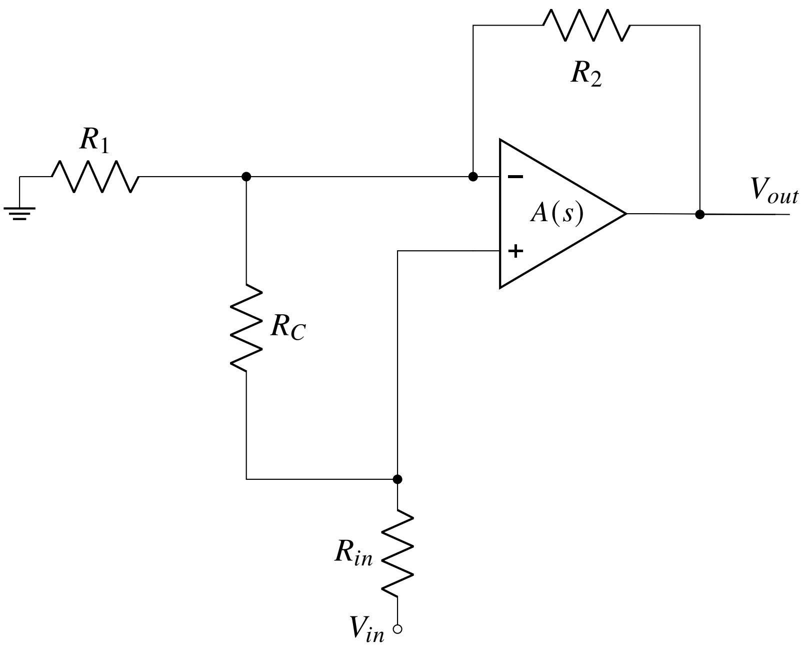


Figure 4.7: R compensation.

Note that as no current, ideally, may flow into R_C, the input impedance of the stage, is, ideally, infinite.

Chapter 5

Noise

Noise transfer wasn't explained in the 2025 class, which is the one I followed. For this reason, I won't be delving too much into it, but I'll explain the basic procedure to obtain the at the output of:
The first step is to compute the transfer function seen by all noise generators in the circuit, which, as the system is assumed linear, will be simply: $T_i(s)$ for the i-th generator
The output power spectral density is the sum of all the power spectral densities at the input, each multiplied by the absolute value squared of the transfer function. This can be proven analytically for a linear system:

$$S_{out}(s) = \sum_{i=1}^{N} S_i(s)|T_i(s)|^2$$

If we substitute $s = 2\pi f j$, we can find $S_{out}(f)$, the *frequency* power spectral density at the output. To find the RMS (Root Mean Squared) value of the noise at the output one simply has to compute the average squared value:

$$E[V_{noise\ out}^2] = \int_0^\infty S_{out}(f)df$$

and then, the RMS value is simply: $RMS = \sqrt{E[V_{noise\ out}^2]}$.

Chapter 6

INA e ISO

6.1 Instrumentation amplifier (INA)

The instrumentation amplifier was introduced to address some of the shortcomings of the voltage subtractor shown in 3.5. In particular, its design allows for a very large common mode rejection ratio (CMRR)[1], meaning that common mode signals have a very low gain (G_{cm}). The INA also offers infinite input impedance, thus allowing it to better sense a voltage signal without affecting it.

6.1.1 Internal structure of the INA

The internal structure of the INA has two stages: an input/gain stage with high differential gain, with infinite input impedance, and a voltage difference amplifier.

[1]the CMRR is defined as $CMRR = G_D/G_{cm}$

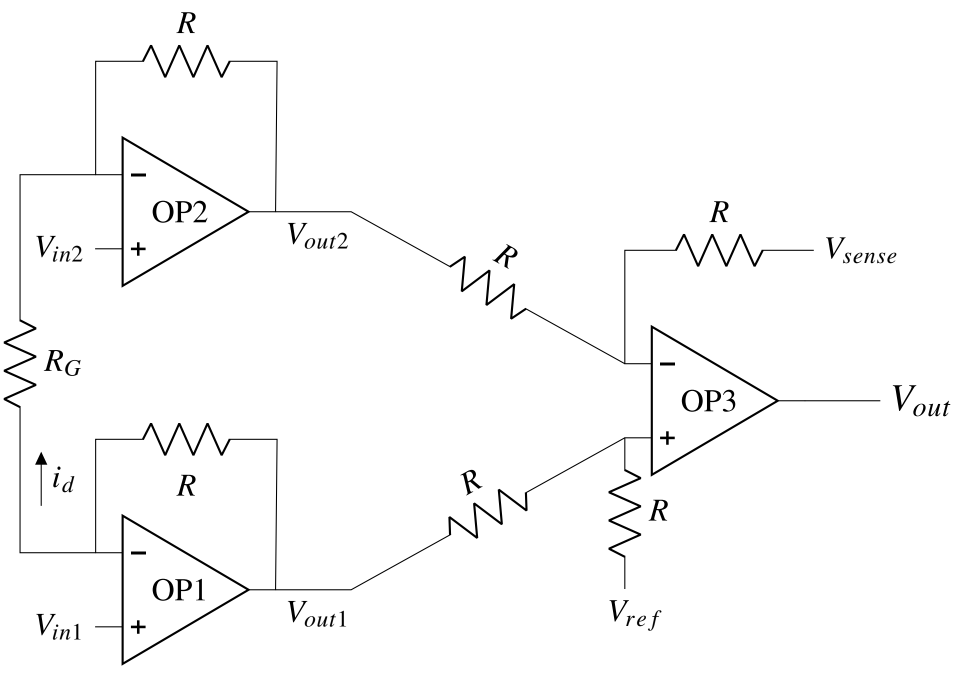


Figure 6.1: Classic three-op-amp Instrumentation Amplifier.

On a differential signal[2], the virtual grounds of OP1 and OP2 transfer the signal across R_G, activating a current $i_d = V_d/R_G$, which then is amplified by the first stage's feedback resistors like a non inverting stage. This results in $V_{out\ 2} = -V_d/2 - V_d\frac{R}{R_G}$ and $V_{out\ 1} = V_d/2 + V_d\frac{R}{R_G}$. As the second stage is just a difference amplifier with gain 1, which computes $V_{out} = V_{out1} - V_{out2} = (1 + \frac{2R}{R_G})V_d$.

On a common mode signal, no current flows into R_G and $V_{out\ 1} = V_{out\ 2} = V_{CM}$. This signal is then removed by the difference amplifier implemented in OP3, whose finite input impedance isn't a problem, as the output impedances of the first stage are zero. Only internal mismatches of R may cause this high CMRR to change, as seen on the chapter on the difference amplifier on this notes.

The V_{ref} pin is used to provide a baseline signal for the output (when no signal is present, that's the output voltage with get), which is very helpful for signals with both polarities that might otherwise go below zero, and can even be used in certain feedback circuits to remove the offset of the INA.
The V_{sense} connection, instead, is where the feedback of OP3 will set the voltage of the INA, and should therefore be connected to the output in most designs.

[2] $V_{in\ 1} = V_d/2, V_{in\ 2} = -V_d/2$

6.1.2 Kelvin connection

The Kelvin connection is used to power a differential sensor far from the circuit board. Parasitic resistances in the wires connecting (and powering) the sensor to the circuit board may produce a voltage drop across the cables, which is interpreted as signal, if those same wires are used to read the voltage across the sensor. This is solved by using different wires to bias the sensor at constant current and to read the differential voltage across it. As no significant current needs to flow across the INA's inputs, there is no voltage drop across the sensing cables, and no error is produced.

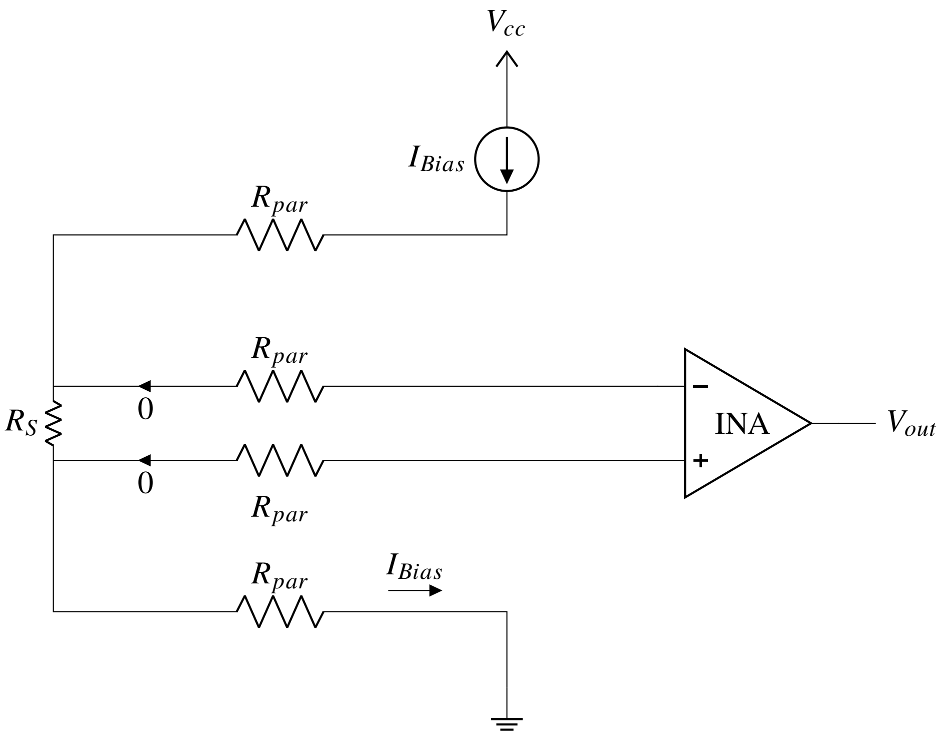


Figure 6.2: Kelvin 4 wire connection. All the voltage drop across R_S is sensed

6.1.3 Guard Pin

Some INA have so called *guard pin* connected to measure the common mode between V_{out1} and V_{out2}. This is achieved by connecting it as follows:

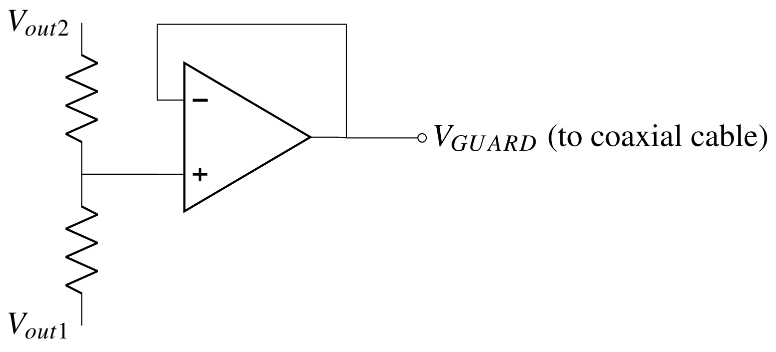


Figure 6.3: Guard connection. This is integrated into figure 6.1

The guard pin is supposed be connected to the cable shield around a coaxial cable at the input of the INA. This prevents the parasitic capacitances of the cable from charging up. This is needed as the stray capacitances touching the two inputs of the ina won't always be matched, which results in slightly different poles and thus a slight common mode signal emerging at certain frequencies.

6.2 Isolation amplifier (ISO)

An isolation amplifier is a device used to transfer an electrical signal between two electrically isolated circuits. This is often useful when dealing with high power applications, where the control logic -often a microcontroller- must sense high voltages while being very delicate. A small spike at the high voltage side might be too much for the microcontroller, easily destroying it. This requires isolation of the two circuits, but it also requires a signal to travel between them to control/sense the high voltage systems.
This can be acomplished in three different ways, all of which are available on the market:

- Optical coupling
- Magnetic coupling
- Capacitive coupling

In the lectures we only studied in detail the first kind, showing an example of an analog buffer implementation using optocouplers. The other devices are more straightforward to use, as it's just enough to follow the recommended implementation schematics in the datasheets.

6.2.1 Optocouplers

Optocouplers are made as four terminal devices, basically corresponding to an LED on one side, and a receiving side which can be modeled as a transistor, whose current is proportional by a factor $\beta \approx 0.01$ to that flowing in the LED. Since the two sides are not in electrical contact, this can be used as an isolation amplifier. One disadvantage is the high spread of β, which requires some clever design solutions to ensure reliable analog transmission of signals.

6.2.2 Digital optocoupling

We might want to transmit and isolate a digital signal using optocouplers. Fig. 6.4 shows a way to achieve this. Note that the first buffer is inverting and is powered by V_1, while the second buffer is not inverting, and is powered by V_2.

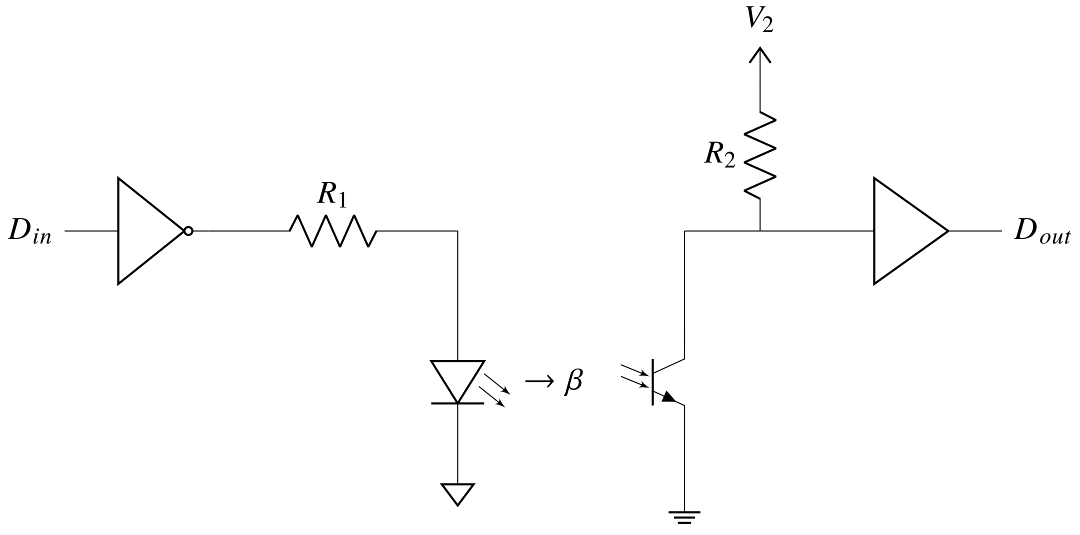


Figure 6.4: Digital isolation buffer

The idea of this stage is to activate the LED when D_{in} is low and the output of the first buffer is high; thus activating current in the transistor and reducing the voltage at the input of the second buffer, whose output goes low. When V_{in} is high, the LED turns off, and the pull up resistor at the output buffer makes that node high.

Note that the resistors must be sized properly for this circuit to work. First of all, R_1, when the first buffer's output is high, must provide the correct current to the LED. If we call this current (found in the datasheet for the optocoupler) I_1, and the voltage drop across the LED (also found in the datasheet for the optocoupler) V_D, the condition that must be verified is that:

$$I_1 = \frac{V_H - V_D}{R_1}$$

From which the correct value for R_1 should be selected.

R_2 should also be large enough to make the node at the output go low when the current across the transistor is βI_1. This requires it to be large. Larger, in fact, than a minimum value given by:

$$V_{LOW}^{max} = V_2 - R_2^{min} \beta I_1$$

Where V_{LOW}^{max} is the highest input value that the output buffer reliably detects as a binary '0'.

6.2.3 Analog optocoupling

We might also want to buffer an analog signal, for instance for measuring a sensor from a high voltage side of a circuit board. To achieve this, a solution that doesn't depend on the exact value of the optocouplers β should be found. Since two optocouplers built on the same chip are likely to have highly matched betas, we can use this to our advantage as in the circuit in figure 6.5.

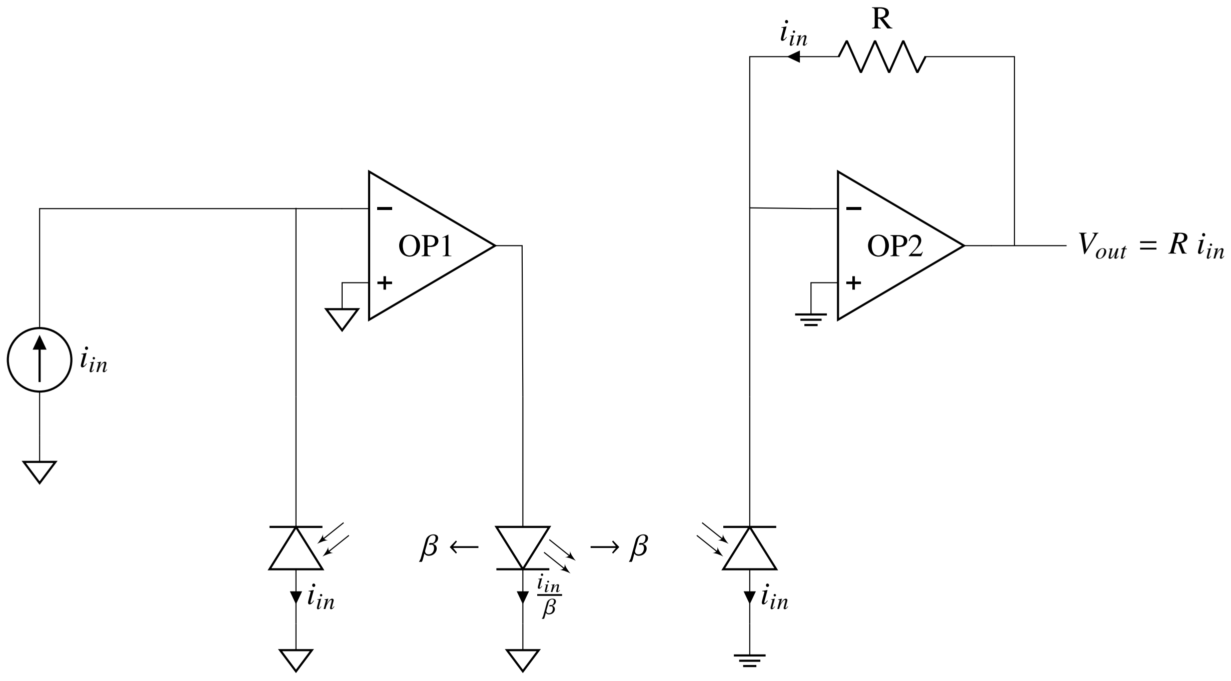


Figure 6.5: Analog isolation buffer with matching

The idea of this stage is to force the input current into the diode at the left. To achieve this, the feedback around OP1 must now make the current into the diode at the center equal to $\frac{i_{in}}{\beta}$. This makes the current into the diode at the right equal to i_{in}. Such a current is then converted into a voltage signal by OP2, which implements a current buffer.

Chapter 7

The current feedback amplifier (CFA)

The current feedback amplifier is an operational amplifier that was designed to be able to control the gain of the stage independently of its bandwidth. This isn't possible using traditional opamp configurations, as $G_{loop}(0)$ usually involves the same terms that appear in the ideal transfer. If the ideal transfer is changed, the loop gain is also changed, leading to a different 0dB cutoff point, and ultimately a different bandwidth.
This trade-off is broken in the CFA by employing a low impedance node at the input. This node, when a signal is presented at the + terminal, tries to replicate it, leading to a current into a resistor which leads to an ideal gain. More on that later. Since when computing the loop gain we should turn off the external generators, the low impedance node acts as a ground, thus shorting a resistor and making it disappear from the loop gain. Hence, it's possible to change the ideal gain and the loop gain independently, making this amplifier widely used in analog applications requiring high speed and gain.

7.1 Small signal model

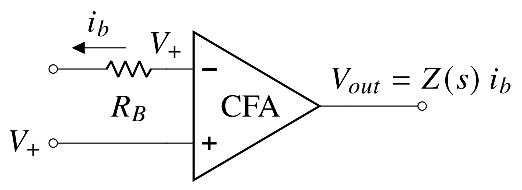


Figure 7.1: Small signal model of the CFA

Basically, the CFA has an internal buffer which reads the voltage at the + terminal and forces it at the minus terminal. The output resistance of the internal buffer is R_B, and the current flowing through it, i_b, is amplified and converted into a voltage signal at the output node. Therefore, the output is sensitive to the current flowing in the input via a transfer which has units of [Ω] and, usually, a single dominant pole, hence: $Z(s) = \frac{Z_0}{1+s\tau_0}$.

7.2 Typical application

The typical application for the CFA is a non-inverting configuration like that shown in figure 7.2.

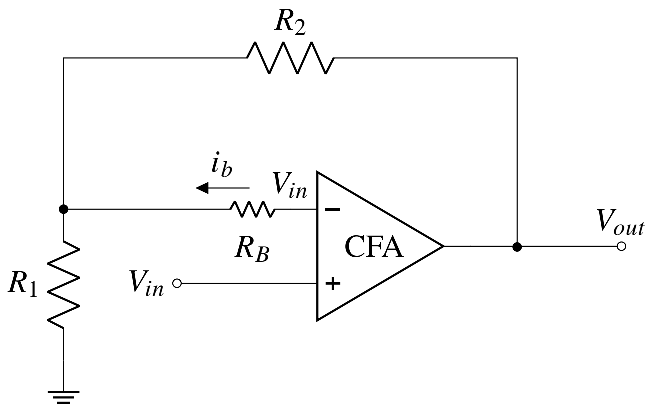


Figure 7.2: Non inverting configuration using a CFA.

Ideally, $R_B \longrightarrow 0$, and V_{in} is perfectly forced at R_1, thus obtaining the ideal gain of a non inverting stage: $G_{id} = 1 + \frac{R_2}{R_1}$. The loop gain is, if we inject a current at the minus node, equal to:

$$G_{loop}(s) = -\frac{Z_0}{1+s\tau_0}\frac{R_1//R_B}{R_1//R_B + R_2}\frac{1}{R_B}$$

Crucially, since R_B is the output resistance of a buffer, it is much smaller than R_1 and R_2, thus $R_B//R_1 \approx R_B$, and:

$$G_{loop}(s) \approx -\frac{Z_0}{1+s\tau_0}\frac{1}{R_B+R_2} \approx -\frac{Z_0/R_2}{1+s\tau_0}$$

As seen in chapter 2 of these notes, the closed loop poles will be at the gain-bandwidth product of this loop gain, which is at:

$$f_P \approx |G_{loop}(0)| \cdot f_0 \approx \frac{Z_0}{R_2}\frac{1}{2\pi\tau_0}$$

This closed loop **doesn't depend on** R_1, which means we can lower R_1 to increase the gain without changingg the bandwidth, as long as the condition $R_1 >> R_B$, which underlies this derivation, is true. This leads to a minimum value of R_1 and thus a maximum achievable gain, without changing the bandwidth, of $\approx 1 + \frac{R_2}{10\,R_B}$.

7.3 Internal layout

The internal layout of the CFA is comprised of an input stage with a source follower (or a push pull) between the + and - nodes, acting as the buffer, and a current mirror carrying this current to the output, where it is amplified and buffered into a voltage signal.

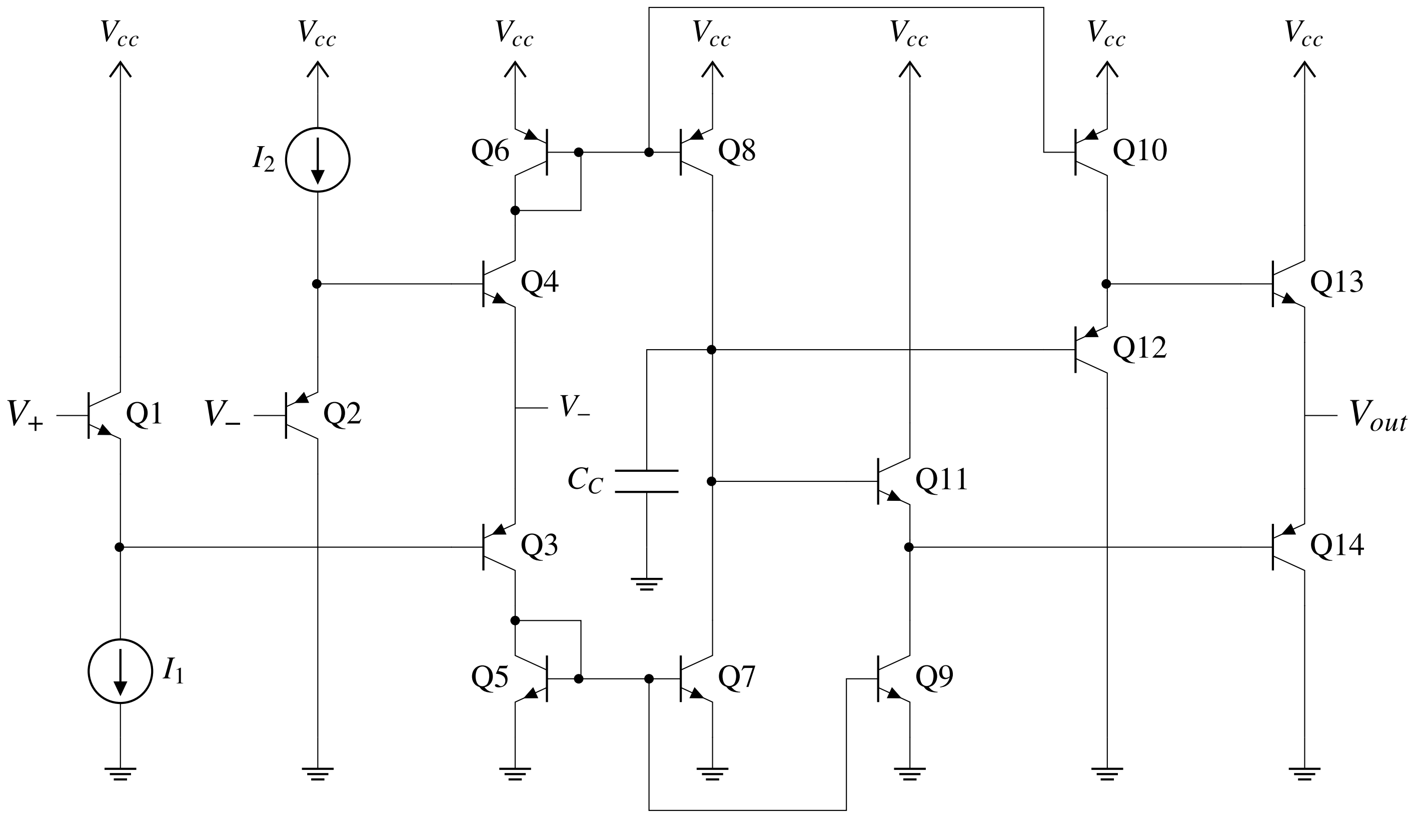


Figure 7.3: Internal layout of a CFA

In this circuit, Q1-Q2 are followers that read the V_+ value, and command the push-pull stage made of Q3-Q4 to force that value onto V_-. This means that $R_B \approx \frac{1}{2\ g_{m\ 3}}$ The mirror made by Q5-Q6 read the current from the -, and commands Q7 and Q8 to carry the same current, which flows into the high impedance nodes obtaining a high voltage gain. This signal is then read by the followers Q11-Q12 and put at the output by the push-pull stage Q13-Q14. To reduce the frequency of the dominant pole, a compensation capacitance is added at the high impedance node.

7.3.1 Slew rate response

Ideally, when a large signal is applied to the + terminal, the followers and push pull replicate this onto the minus terminal, thus activating a large current into R_1 of figure 7.2. This current is then mirrored by Q5-Q6-Q7-Q8 and flows into the compensation capacitance. Note that at first the output node is stuck, as the voltage across C_C is still zero. This means that forcing V_{step} to the + terminal also activates a current in R_2, which is also mirrored into C_C. This leads to an initial slew rate of

$$SR \approx \frac{V_{step}}{(R_1//R_2)C_C} \approx \frac{1V}{1\ k\Omega\ 1\ pF} \approx 10^9\ \frac{V}{s} = 1\ \frac{V}{ns}$$

Which allows for a very large bandwidth. For a 2 V sinusoid, we'd have about $f_{max} = \frac{SR}{2\pi V_p} \approx 80$ MHz, all while still achieving a good gain.

In reality, the transient response of the CFA in 7.3 is also limited by the Q1-Q2 input pair. For instance, a large swing up of V_+ is expected to turn off Q2 and increase the current in Q1, thus leaving the base of Q4 only charged up by I_2, while the base of Q3 is more quickly charged by the highly active Q1.

Chapter 8

OTA e Norton

8.1 Operational transconductance amplifier

The OTA is an operational amplifier that takes an input differential voltage signal and outputs a current proportional to it via its gain, which has units of $S = \frac{1}{\Omega} = \frac{A}{V}$[1]. It is therefore characterized by the equations: $I_{out} = G_m(V_+ - V_-),\ i_+ = i_- = 0$.

8.1.1 Basic structure

The basic structure of the OTA is widely known and studied. It consists of a differential stage with mirrors to add up the differential currents generated by the stage. The mirror can either be integrated into the load, in a compact single stage as the one shown in figure 8.1, or more mirrors could be used to decouple input and output dynamics.

[1] S = Siemens

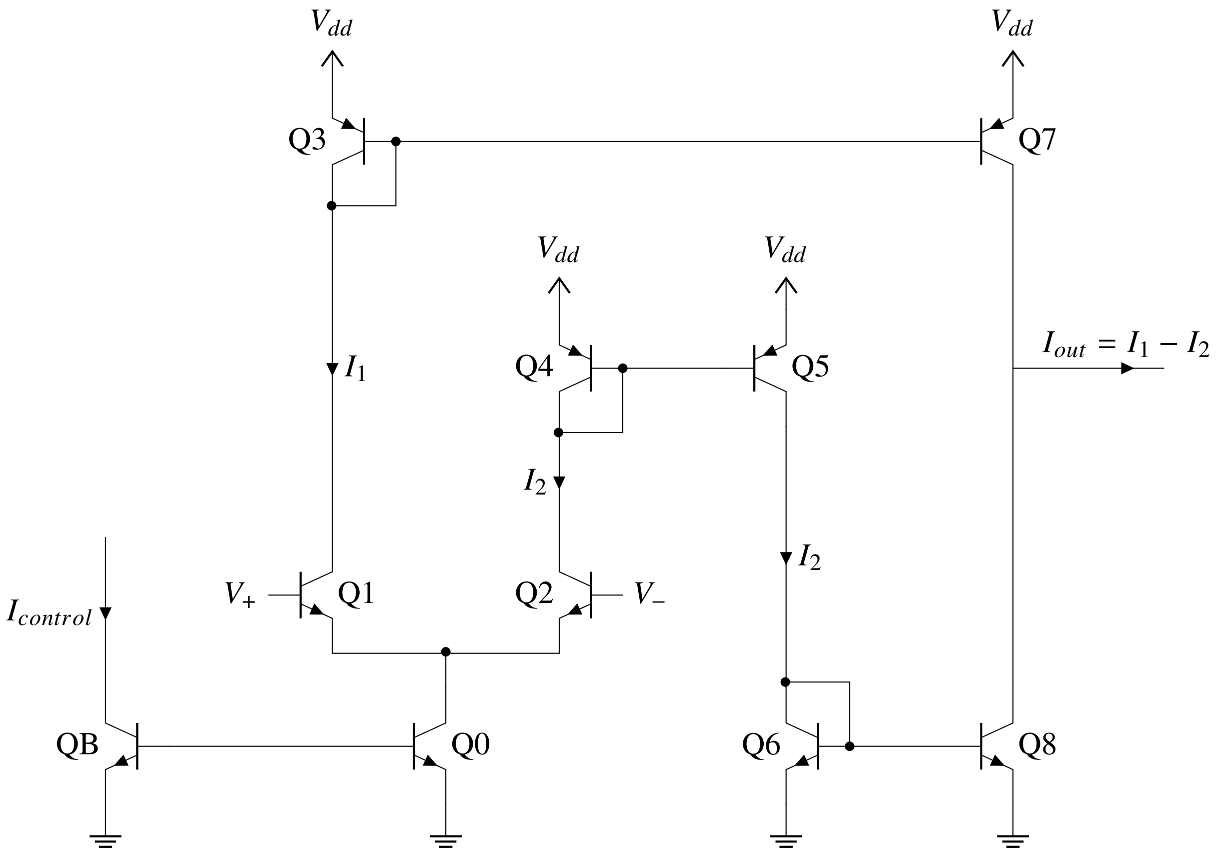


Figure 8.1: Basic structure of the OTA.

This design has a further input than the basic OTA: the tail generator, which provides the bias to the whole OTA, is connected to a current mirror which allows the user to select the bias current of the whole stage, I_0, which is then split into the two branches leading to a transconductance of the two input transistors equal to $g_{m\ 1} = \frac{I_0/2}{V_{Th}}$. The current is selected by placing a resistor R_B in series to QB, connected to the power supply, and it allows to select $I_{control} = \frac{V_{dd} - 0.7\ V}{R_B}$. This, in turns, allows to select the current gain of the whole OTA by modulating R_B in the previous equations, by keeping in mind that the transconductance of the whole OTA on a differential signal is the same as that of Q1. Note that the mirroring ratio between QB and Q0 isn't always one, which can lead to an I_0 different from $I_{control}$. For instance, if the gain of the mirror is 2, we'll have that each one of the two differential pair transistor's gets a current equal to $I_{control}$, leading to an overall transconductance of $g_m = \frac{I_{control}}{V_{Th}}$. This choice is very popular and underlies all the problems involving OTA's seen in the course.

When it comes to the frequency response, note that the circuit in figure 8.1 has only a single high impedance node at the output, which means that the parasitic and load capacitances on that node will produce a single low frequency pole.

Also, keep in mind that due to its high output impedance, the OTA is best suited to work when piloting a capacitive load. When a resistive load is used, the loop gain of the OTA will degraded, leading to a non ideal feedback behavior. Another issue with OTA's is the nonlinearity of their gain, which, for large swings of the input differential signal, will lead to a variation of the transconductance, which can only be taken as constant in a linear, first order, analysis. To address this issue, which can be relevant when the amplifier is used in open loop to directly drive a load, a solution has been devised which we shall see in the next section.

8.1.2 Linearization

To improve the linearity of the OTA's response on a differential signal, a solution has been devised, and it's shown in figure 8.2. This circuit can, if the I_D and $2I_D$ generators are turned off, become equivalent to the standard differential stage. Commercial opamps have both modes of operation, which can be turned on or off by a control pin.

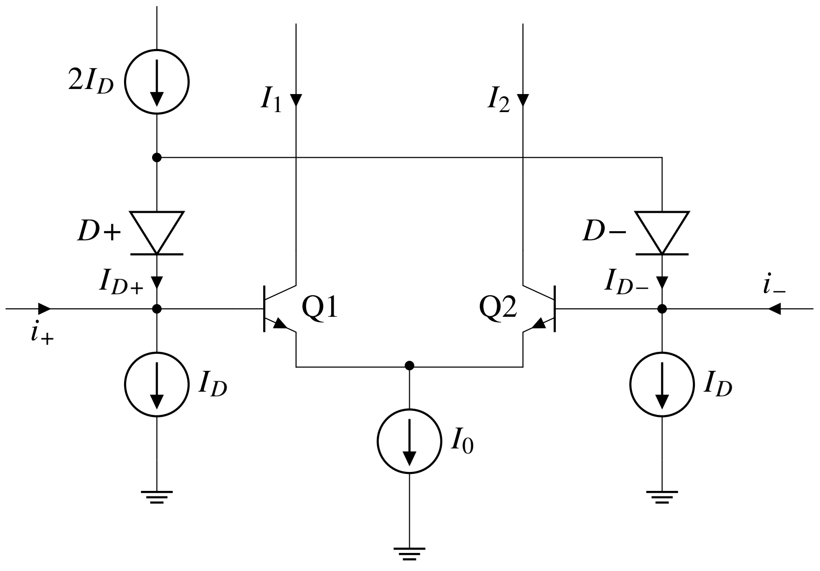


Figure 8.2: Linearization circuit for the OTA's differential pair.

The idea of this modification is to create a closed loop of voltages with the diodes D+, D- and the V_{BE} of Q1 and Q2. This loop introduces a KVL like

$$
\begin{aligned}
V_{BE\ 1} + V_{D+} &= V_{BE\ 2} + V_{D-}, \text{Which implies} \\
\ln(I_1/I_S) + \ln(I_{D+}/I_S) &= \ln(I_2/I_S) + \ln(I_{D-}/I_S), \text{ and} \\
I_1 \cdot I_{D+} &= I_2 \cdot I_{D-}
\end{aligned}
$$

This relation links the currents in the diodes with those of the transistors. The current generator at the top forces a further condition on the diode's currents: $I_{D+} + I_{D-} = 2I_D$, and the input and output nodes further force the relations: $i_- + I_{D-} = I_D$ and $i_+ + I_{D+} = I_D$.

We also know that $I_1 + I_2 = I_0$, the tail generator. These equations have 4 unknowns and can be solved to obtain I_1 and I_2 as a function of i_- and i_+. The result is:

$$I_1 = \frac{I_0}{2I_D}(I_D + \frac{i_+ - i_-}{2})$$
$$I_2 = \frac{I_0}{2I_D}(I_D + -\frac{i_+ - i_-}{2})$$

From which, we can compute the output current of the ota as seen in figure 8.1, leading to:

$$I_{out} = I_1 - I_2 = \ldots = \frac{I_0}{2I_D}(i_+ - i_-) \tag{8.1}$$

Equation 8.1 tells us that the OTA's output, when linearization is active, is linearly dependent on the differential input current. This allows us to remove any distortion, albeit at a price of a severely reduced gain, which, in current, is now just of a few tens. To apply a voltage input, we can simply note that V_+ and V_- must be at about the same potential due to the diodes. This means that a differential signal placed across them, with resistors as in figure 8.3, lead to an input differential current proportional to it. In this example, a current equal to $\frac{V_d}{2\,R_d}$ flows into the generator, leading to $i_+ = \frac{V_d}{2\,R_d}$ and $i_- = -\frac{V_d}{2\,R_d}$, thus obtaining $i_{out} = \frac{I_0}{2I_D}(i_+ - i_-) = \frac{I_0}{2I_D}(\frac{V_d}{R_d})$, which is still somewhat lower than what can be obtained with a non linear OTA.

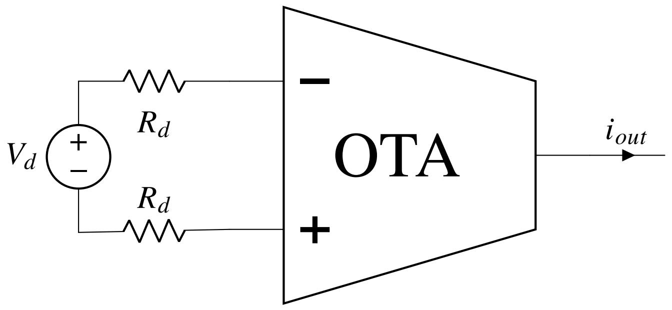


Figure 8.3: Using a linearized OTA

8.2 Norton amplifier

8.2.1 Small signal model

The small signal model of the Norton amplifier is shown in figure 8.4. This amplifier forces the voltage at its central input pin, V_M, onto the other two terminals, and produces a current output proportional to the difference betweenn the currents flowing into the + and - inputs. Ideally, the amplifier should act to reduce this difference down to zero, making the two currents equal. The advantage of this operational amplifier is having low impedance nodes at the input, which allow for larger bandwidth than conventional opamps. Since the current gain A_i has only high frequency poles, its bandwidth should be limited by placing a capacitor at the high impedance output node. In practice, a parasitic capacitance loading that node already exists, and it is responsible for the pole.

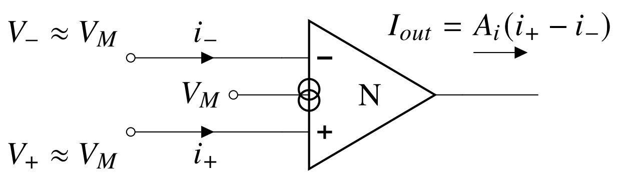


Figure 8.4: Small signal model of the Norton amplifier.

8.2.2 Input stage

The input stage of the Norton must both sense the voltage V_M, buffer it to the two terminals, and sense the two currents to compute their difference. This is challenging and requires a specialized structure, shown in figure 8.5. Note that the diodes D1 and D2 are made by diode-connected (shorting the base and collector) bjt transistors, which ensures their matching with the other transistors in the stage.

The working principle is the so called *translinear principle*, which can be explained by writing the KVL on the loop made by Q1-Q3 and D2-D1. Please note that a BJT transistor has an I-V relation of the kind $I_E = I_S\, e^{V_{BE}/V_{Th}}$.[2] Inverting it, we get $V_{BE} = V_{Th} \cdot \ln(I_E/I_S)$. The KVL reads as:

$$\begin{gathered} V_{EB\ 3} + V_{BE\ 1} = V_{D1} + V_{D2} \\ V_{Th} \cdot \ln(I_1/I_{Sp}) + V_{Th} \cdot \ln(I_3/I_{Sn}) = V_{Th} \cdot \ln(I_0/I_{Sp}) + V_{Th} \cdot \ln(I_0/I_{Sn}) \\ I_1 \cdot I_3 = I_0^2 \end{gathered}$$

When there is no input current, $I_1 = I_3$, and the two currents must be equal to I_0. The bias is thus fixed by the translinear principle at the input KVL loops. This also means that the V_{BE} must be equal, as the I_S of the diodes and transistors are matched. The same reasoning, applied at the - input, allows us to say that $I_2 \cdot I_4 = I_0^2$, and therefore that:

$$I_1 \cdot I_3 = I_2 \cdot I_4 \tag{8.2}$$

This is the ultimate expression of the translinear cell, which is sometimes used to multiply analog signals, as it provides an easy way to achieve this sort of operation in analog systems.

The input current i_+, **on a small signal**, is equally split betweenn the two $1/g_m$ impedances seen at Q1 and Q3, and the same goes for the i_- current. If $i_+ = i_-$, this leads to two

[2]where $V_{Th} = \frac{kT}{q} \approx 25.8$ mV

identical signal currents flowing into the two branches, which the following stage, that must compute their difference, should eliminate.
If i_+ and i_- are not the same, the two currents will be unmatched, leading to a net output current that is not zero. In general, this results in a small signal and bias current of:

$$I_1 = I_0 - i_+/2$$
$$I_2 = I_0 - i_-/2$$
$$I_3 = I_0 + i_+/2$$
$$I_4 = I_0 + i_-/2$$

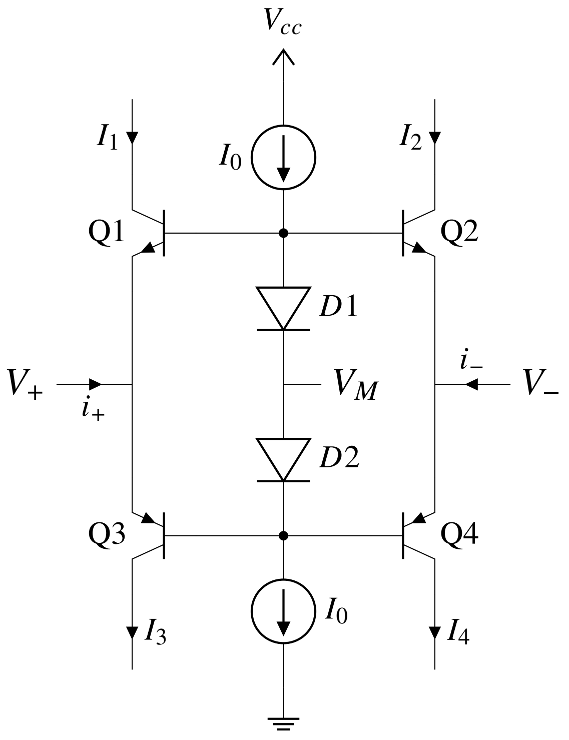


Figure 8.5: Input stage of the Norton amplifier

8.2.3 Second stage

The second stage is composed of several current mirrors aimed at computing the difference betweenn the two signal currents. Note that a single current mirror copying both currents into the same path is not enough, as that would lead to their sum into the output node. To compute their *difference*, two mirrors should be employed to reverse one of the currents before adding them up at the output node. This solution, which should be doubled both above and below the input stage of figure 8.5 is shown in figure 8.6 for the difference of just two currents. Note that by making the Q1-Q2 and the Q5-Q6 mirrors asymmetrical, a current gain of a few tens may be obtained. This doesn't allow the Norton amplifier to act as ideally as the regular voltage operational amplifier, and in most cases the full model, not the ideal one, should be used for computations[3].

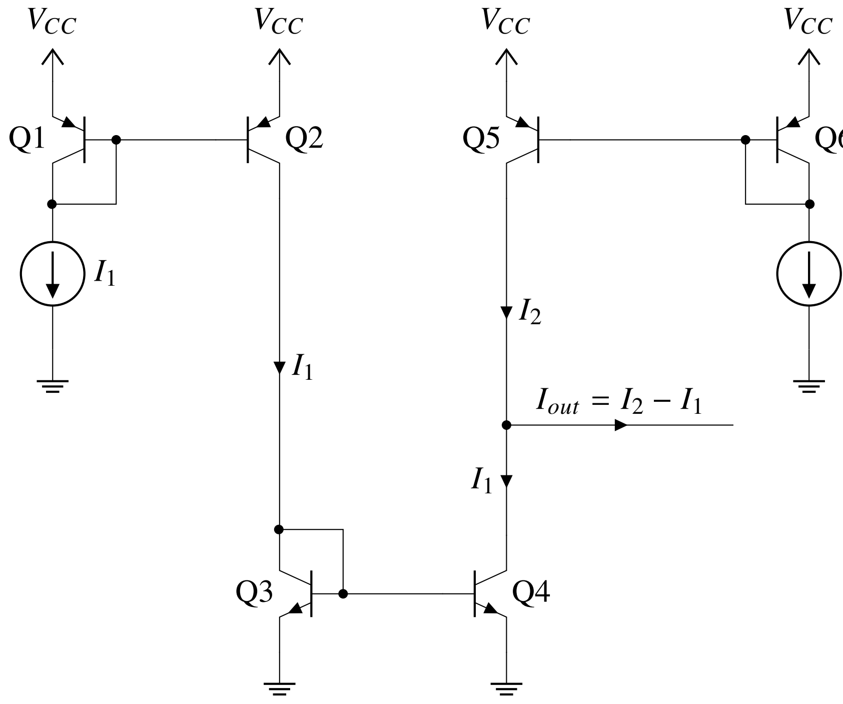


Figure 8.6: Second stage of the Norton amplifier. This stage just computes the difference betweenn two currents I_1 and I_2.

This stage is quite simple, it uses the mirror Q1-Q2 and the mirror Q3-Q4 to invert (or bring below) I_1, which is summed to I_2 at the output node.

[3]use the model with finite A_i, and consider that $i_+ \neq i_-$.

Chapter 9

Sampling

Sampling is the act of taking points on an analog signal and converting them to digital numbers. It's essential every time an analog information, like that from a sensor, is to be sent to a computer for storage or further processing. As every sample takes up valuable memory in a computer, we might be interested in reducing the number of samples needed to correctly represent an analog signal in a digital system. We might ask ourselves if there is a lower limit to this number.

9.1 The Shannon theorem

The Shannon sampling theorem tells us the minimum sampling rate at which an analog signal must be sampled in ordered to preserve its information. It is illustrated in figure 9.1. It states that *The minimum sampling frequency is twice the bandwidth of the signal*. This is due to the effect of sampling on the spectrum of a signal: when a continuous time signal is sampled at a frequency f_s, its spectrum becomes periodic, in the frequency domain, with a period f_s, as shown in figure 9.2. Sampling at frequencies below the Shannon limit results in *aliasing*, which involves a distortion of the Fourier spectrum of the signal due to higher frequency components being indistinguishable from lower harmonics. This phenomenon is known as *folding*, and it's illustrated in figure 9.3. To ensure this limit is always met, every acquisition chain must include a so-called *anti-aliasing* filter, which is basically a low pass filter that removes all harmonics above $f_s/2$.

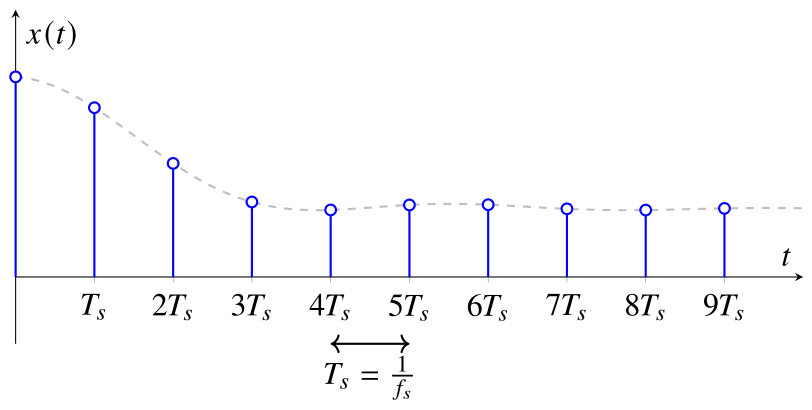


Figure 9.1: Time-domain representation of sampling an analog signal with period $T_s = 1/f_s$.

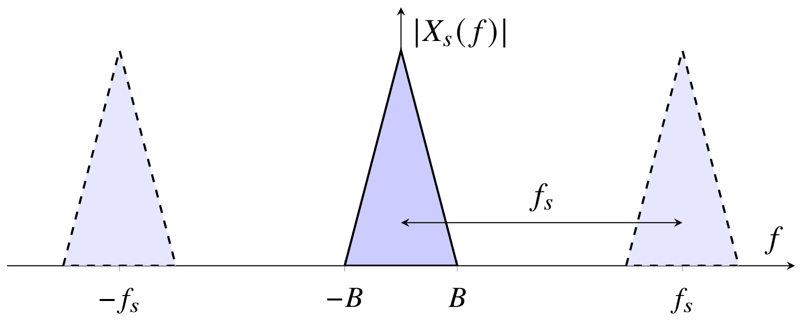


Figure 9.2: Periodic spectrum showing the relationship between bandwidth B and sampling frequency f_s.

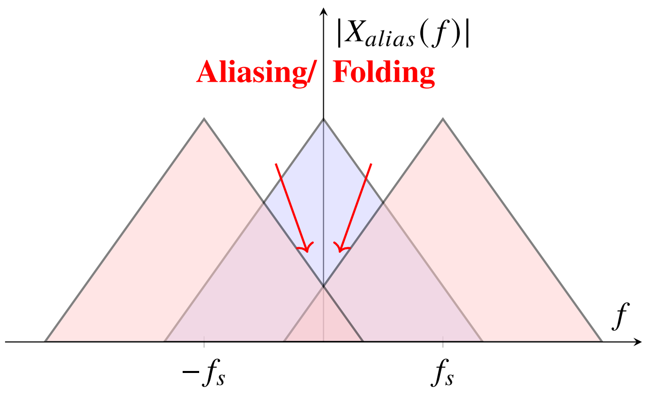


Figure 9.3: Aliasing effect when f_s is below the Shannon limit. The triangles overlap distorting the spectrum.

9.2 Reconstruction and equalization

In order to figure out the value of the signal in between samples, which is often needed, we must reconstruct it. Note that the sampling process in figure 9.1 assumes that the signal is sampled instantaneously, only acquiring a single point in time and reconstructing it according to the ideal Shannon reconstruction formula, which is very long and not needed here. A faster way to reconstruct the signal is to just assume it is constant for until the next sample, as shown in figure 9.4.

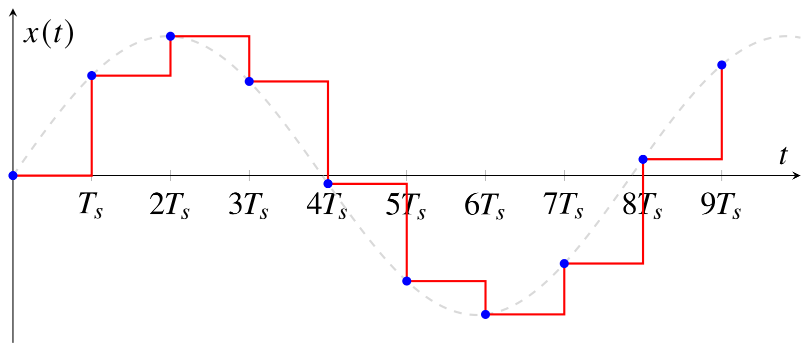


Figure 9.4: Simplified reconstruction. The signal is held constant at the sampled value for the duration of the period T_s, resulting in a staircase approximation.

The problem with this reconstruction method is that the signal is distorted. In the time domain, this is like convolving the sampled signal by a rectangle of width T_s. In the frequency domain, this corresponds to a multiplication by the Fourier transform of the rectangle, which is $\frac{\sin(\pi f T_s)}{\pi f T_s}$. In order to avoid this problem, the signal must be pre-distorted before the reconstruction by a filter with the opposing frequency response, which is $\frac{\pi f T_s}{\sin(\pi f T_s)}$. This process is called *equalization*, and it's often needed in high fidelity systems, like audio, to preserve the signal.

9.3 DFT

I think it's important to know how to compute the spectrum of a provided signal, so I want to add to these notes the definition of the Fourier transform for a discrete signal. This definition is easy to implement in code and something everyone working on signals should know. The discrete Fourier transform (DFT) of a sampled signal $x[n]$ of length N is defined as:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}nk} \quad k = 0, 1, \ldots, N-1 \tag{9.1}$$

Where:

- $x[n]$ is the input signal in the time domain.
- $X[k]$ is the transformed signal in the frequency domain.

- $e^{-j\frac{2\pi}{N}nk}$ represents the complex sinusoidal basis functions (roots of unity).

Applying this definition allows us to see the spectrum of the sampled signal.

Chapter 10

FDA e Sample and Hold

10.1 Fully differential amplifier

10.1.1 Small signal model

The FDA is designed to work on *differential signals*. A differential signal, which is usually the difference between two voltages, is a more reliable way of representing information in an analog processing chain, as it doesn't depend on a common reference voltage (GND), which could suffer from small variations between different amplifiers. The signal is also immune to variations of the power supply, as they affect both lines in the same manner. In general, all noise sources which are common to both lines, including external disturbances, are rejected. To amplify a differential signal into another differential signal, the fully differential stage of figure 10.1 was introduced. The stage forces $V_{out+} - V_{out-} = A_0(V_+ - V_-)$, while the common mode voltage of both output nodes, which will be their voltage when no signal is applied, is selected by the V_{CM} input. This was done to allow selecting a proper CM voltage to bias the subsequent stage.

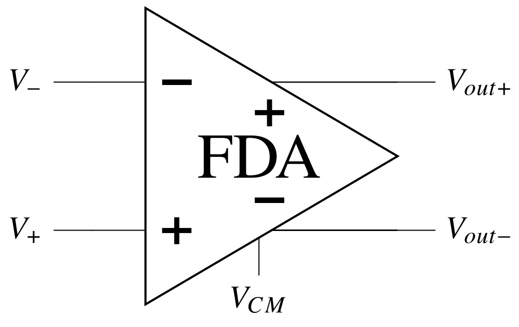


Figure 10.1: Fully differential amplifier.

10.1.2 Example of usage

Figure 10.2 shows an example of an amplifier employing the FDA. The amplifier's ideal feedback tries to short its two inputs, meaning that a current $i_d = \frac{V_{in}}{2\,R_1}$ arises, and flows into both feedback branches.

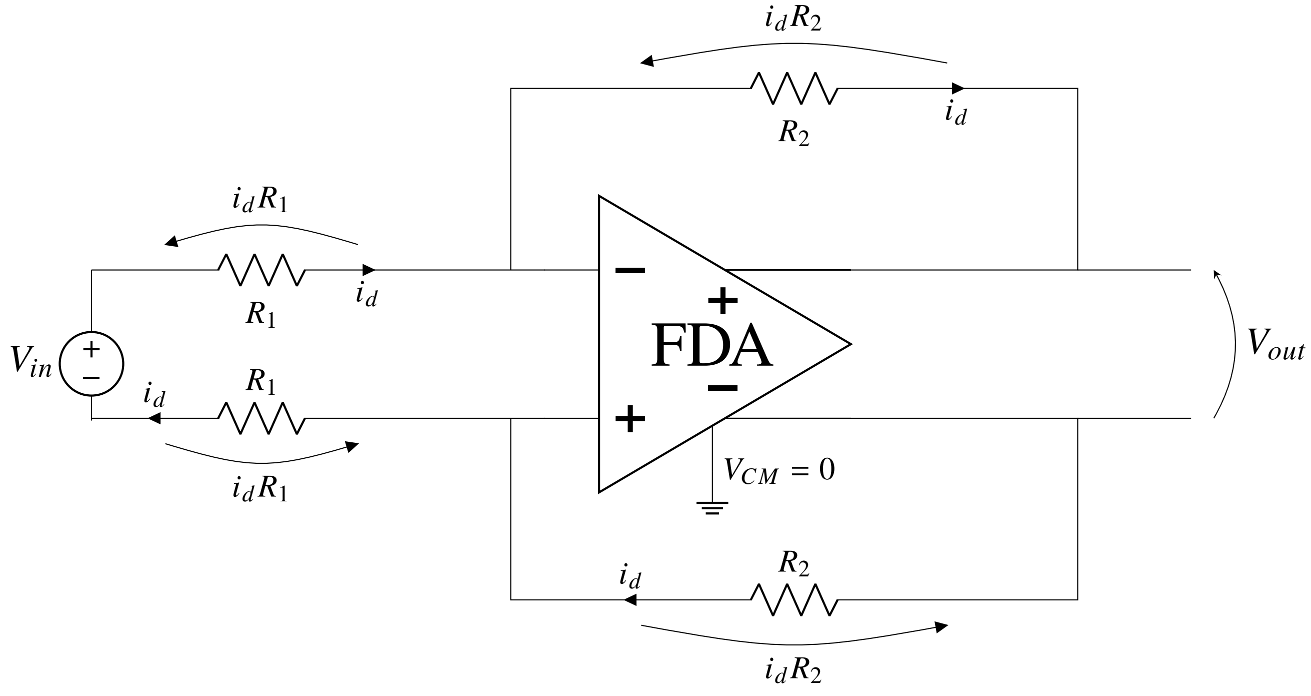


Figure 10.2: Basic amplifying stage using the FDA.

The KVL around the entire device is: $i_d R_2 + V_{out} + i_d R_2 + i_d R_1 - V_{in} + i_d R_1 = 0$, which implies $\frac{V_{in}}{2\,R_1} R_2 + V_{out} + \frac{V_{in}}{2\,R_1} R_2 + \frac{V_{in}}{2\,R_1} R_1 - V_{in} + \frac{V_{in}}{2\,R_1} R_1 = 0$. Solving for V_{out} we get $V_{out} = -\frac{R_2}{R_1} V_{in}$, which is the same as the inverting configuration with the single-handed opamp.

10.1.3 Internal layout

The internal layout of the FDA must create a differential signal from the two inputs, and amplify it across different paths, with a low impedance buffer at the output. This while maintaining a reliable common mode bias that is selected from the V_{CM} pin.

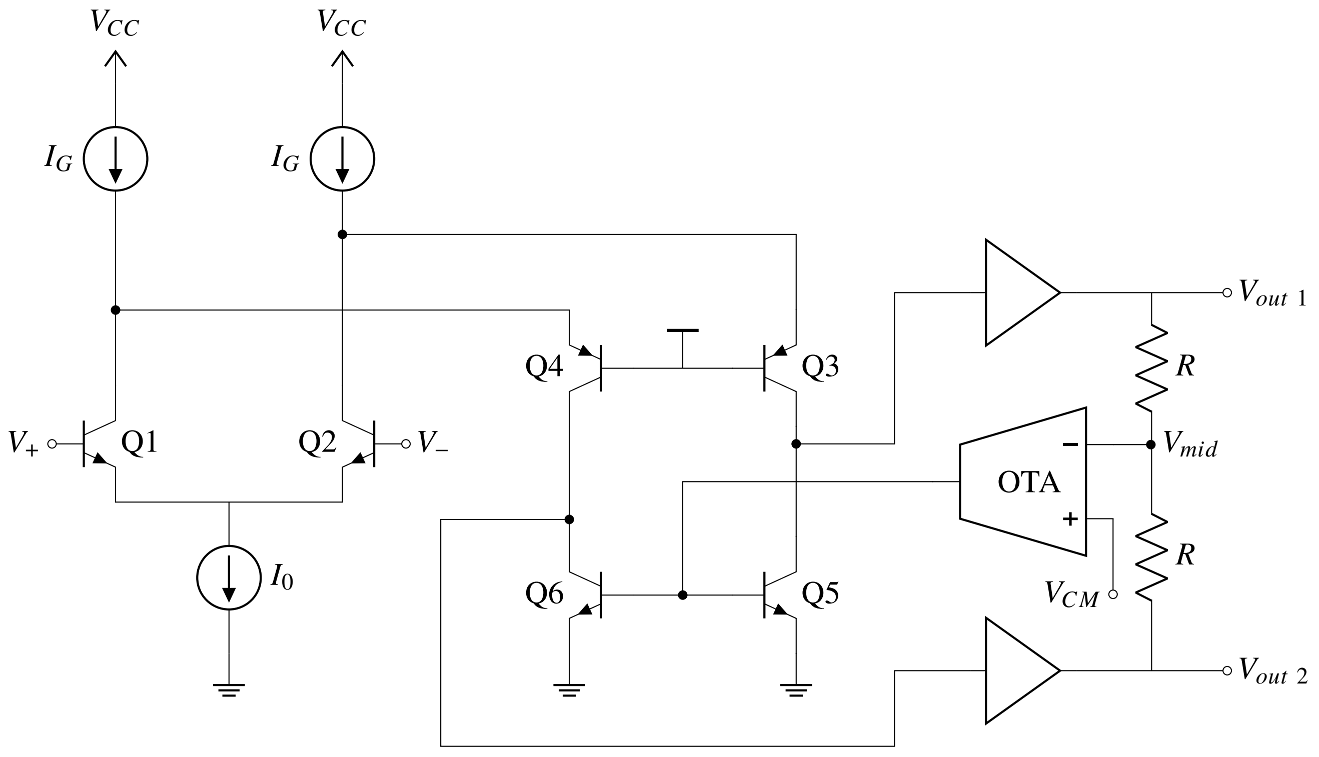


Figure 10.3: Internal schematics of the FDA.

This circuit is composed of a differential pair Q1-Q2, which generates a differential current that flows into the Q4-Q3 current buffers and obtains a high gain into the high impedance collectors of Q6-Q5. The buffers then take these voltages and force it at the output node. A common mode feedback network measures the output common mode voltage, called V_{mid} and attemps to force it to be V_{CM} by acting on the current flowing into the bases of Q6 and Q5. Note that in a pratical implementation a better way to measure the common mode voltage at the output would be with another differential pair, which would show very high impedance. This would work because the emitter node of a differential pair moves $\approx$ by the common mode at the input.

Note the the bias to Q3-Q4 should be provided by a voltage, as the current flowing into them is already set by the difference of $I_G - I_0/2$. Any voltage set at their base will produce a variation of their emitter potential to allow their current to remain constant.

10.2 Sample and hold (S&H)

The sample and hold circuit is used to acquire an analog voltage and save it on a capacitor while it is being converted by an analog to digital converter (ADC). This system must therefore be able to both quickly acquire the input voltage, with the minimum possible error, and then keep it constant for as long as possible during the conversion phase. This highlights two different modes of operation: Sample and Hold. Any design for a S&H

stage must be able to achieve this double operation, usually under the control of an external digital signal.

10.2.1 Basic design

The basic design of a sample and hold circuit is shown in figure 10.4. It consists of a MOS transistor (M1) being used as a switch, a hold capacitor C_H that is charged up to the desired voltage, and a voltage buffer to read it without discharging the capacitor. Note that to prevent discharge due to the bias currents of the opamp, it is preferred to use a MOS differential input pair. An opamp with a BJT input pair will show a non negligible bias current which will, in time, produce an error on the capacitor. Also note that the offset of the opamp will inevitably be carried out at the output, producing an error which is usually in the order of a few mV, thus limiting the maximum resolution obtainable by this stage. This means that it makes no sense to select an ADC whose resolution is much better than the errors introduces by the S&H stage. As these errors impose such significant design constraintt, we'll study them in detail in the following sections.
The error due to a static current I_B flowing into a capacitor C_H for a time dt, in any case, is given by $dV_H = \frac{I_B}{C_H}$, which reveals that a large C_H is less prone to this issue.
Usually, these error pose certain design constraint depending on the maximum error we are willing to accept. Usually, for an ADC with n bits, we take $dV_H^{max} \approx \frac{V_{ref}}{2^{n+1}}$, which ensure that the total error is less than half the resolution of the ADC, which is $\frac{V_{ref}}{2^n}$.

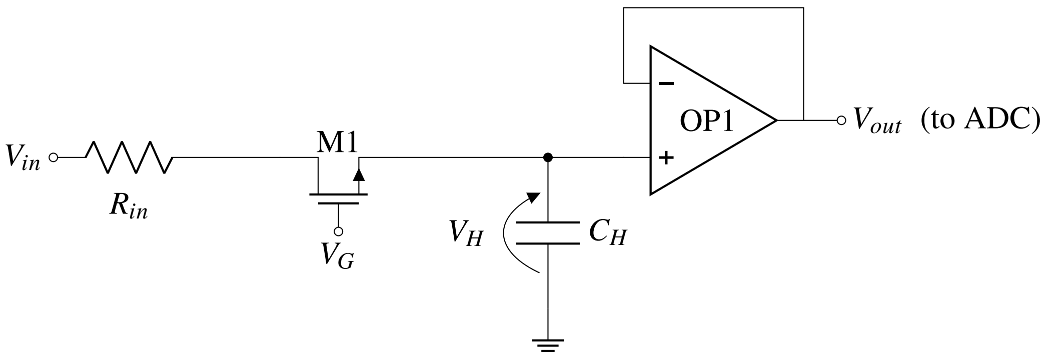


Figure 10.4: Basic sample and hold circuit.

Note that the output resistance of the previous stage, R_{in}, in series to the R_{ON} of the MOS, will lead to an RC circuit-like behavior, which will charge the capacitor following: $V_H = V_{in}(1 - e^{-t/\tau})$, with $\tau = C_H(R_{in} + R_{on})$. To find R_{on}, we can remember that the current of the transistor in the ohmic region is $I_{DS} \approx \mu_n C'_{ox} \frac{W}{L}\ (V_{GS} - V_T)V_{DS}$. R_{ON} is thus defined as $R_{ON} = \frac{dV_{DS}}{dI_{DS}} = \frac{1}{\mu_n C'_{ox} \frac{W}{L}\ (V_{GS} - V_T)}$, which, of course, won't be constant as the capacitor charges up and modifies the voltage at the source of the transistor. In a first order analysis, however, we can neglect this effect and consider $R_{ON} \approx constant$ during this process.

10.2.2 Charge injection

The charge injection phenomenon is often highly misunderstood. It originates from the fact that after the transistor has been turned off ($V_{GS} < V_T$), the C_H is no longer driven by the V_{in} generator: they are not connected. This means that, **starting from that point**, the voltage on the capacitor is susceptible to disturbances due to the charge coming from the transistor's channel, which we can model as its C_{GS} parasitic capacitance. If the **total** variation of V_G used to turn off the transistor is ΔV_G, this effect will only be caused by the amount of variation of the gate occurring **after** the transistor has been turned off, which, since the voltage across the capacitor (which is at the source of the transistor) is around $V_H \approx V_{in}$, will be $\Delta V_G - V_T - V_{in}$. The subsequent variation to the voltage across C_H can be written according to the capacitive partition formula, which is

$$\Delta V_H = \frac{C_{GS}}{C_{GS} + C_H}(\Delta V_G - V_T - V_{in}) \tag{10.1}$$

Equation 10.1 can be derived by considering that the total charge between the two capacitors, once the MOS is turned off, is trapped and cannot change. If the voltage across the series is changed by $\Delta V_G - V_T - V_{in}$, the variation across just C_H will be exactly what shown above, since the voltage across a capacitor is linked to its charge by the capacitance, as $dQ = C\ dV$. If we set $dQ_{GS} = dQ_H$, we get $dV_{GS}C_{GS} = dV_H C_H$, and since it must be that $dV_{GS} = dV_G - dV_H$, we get $(dV_G - dV_H)C_{GS} = dV_H C_H$, and therefore $dV_H = dV_G \frac{C_{GS}}{C_{GS}+C_H}$, which is just equation 10.1. Figure 10.5 illustrates this effect.
This equation reveals that to reduce the effecs of charge injection, a large C_H is needed, at the cost of a slower charging up during the sampling phase and therefore a lower maximum sampling rate. It also shows that the error depends on the value of V_{in} saved on C_H, which makes it non-constant and thus harder to remove in post processing.

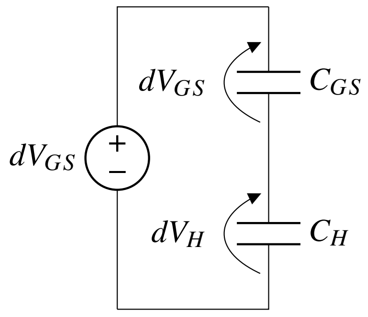


Figure 10.5: Capacitive voltage divider (Partition) circuit.

10.2.3 Feedthrough

Feedthrough is similar to the issue of charge injection described in detail above. Basically, a variation of V_{in} equal to ΔV_{in} after the transistor has been turned off will lead to a variation of the charge across the parasitic C_{DS} of the transistor. The capacitive voltage partition equation still applies, yielding an expression very similar to equation 10.1: $\Delta V_H = \frac{C_{GS}}{C_{GS}+C_H}(\Delta V_{in})$. Note that this effect doesn't depend on the voltage stored across C_H, but it does depend on what the input is doing, which may be unpredictable, albeit very limited. In fact, if the S&H circuit is fast enough compared to the signal's bandwidth, this variation is usually negligible.

10.2.4 Aperture delay and jitter

Aperture delay

Rather confusingly, this phenomenon has nothing to do with the MOS opening up (turning off). In fact, it's caused by time it takes for the mos to close (turn off), between the sample and the hold phase. If we imagine a variation of V_G that is linear in time, starting from V_H and going to V_L in a time T_{HL}, it is $V_G(t) = V_H - \frac{V_H - V_L}{T_{HL}}t$. The delay can be estimated as the time needed by $V_G(t)$ to reach the turn off point of the transistor, which is for $V_G - V_H = V_T$, with $V_H \approx V_{in}$. Let's call it T_d.
During T_d, which can be computed as seen above, the input might change, leading to a variation in the value V_H stored on the capacitor. This will be happening during the conversion of the ADC, and is thus highly undesirable. We can estimate the magnitude of this effect as simply $dV_H \approx \frac{dV_{in}}{dt}T_d$. Assuming a sinusoidal input signal $V_{in} = A_0 \sin(2\pi f t)$, and taking the highest possible $\frac{dV_{in}}{dt}$, we get $dV_H = A_0 2\pi f T_d$, which will impose a constraint on the maximum allowed frequency of the input signal depending on the maximum error we are tolerating.

Aperture time jitter

Since the turn off V_G of the transistor is $V_G - V_H = V_T$, with $V_H \approx V_{in}$, we have that the time needed for the MOS to close properly (T_d) depends on the input signal. Furthermore, the threshold voltage of the transistor won't be exactly the same for all possible production samples of the device. This means that the exact time needed for the transistor to close will change somewhat from one unit to the next. As seen above, this leads to an increase (or decrease) of T_d, which we can estimate by slightly nudging V_T in the equation seen above for T_d. This will lead to an error due to a variation of V_{in} that also depends on the spread of the threshold voltages of the transistor. This effect, however, is usually negligible compared to the error due to delay itself, which can still be computed considering a worst case value of $V_T \approx V_{T\ nominal} + 3\ \sigma(V_T)$ to take into account this effect.

10.2.5 Charge injection compensation

A circuit solution to compensate the charge injection effect is shown in figure 10.6. The idea of this stage is to duplicate the charge injection effect in another set of capacitor and transistor, but in an opposing direction. When V_G is moved down resulting in a dV_H across both C_H, the resulting output variation is null. As long as the pair is matched well, this compensates for the variation of V_H due to charge injection.

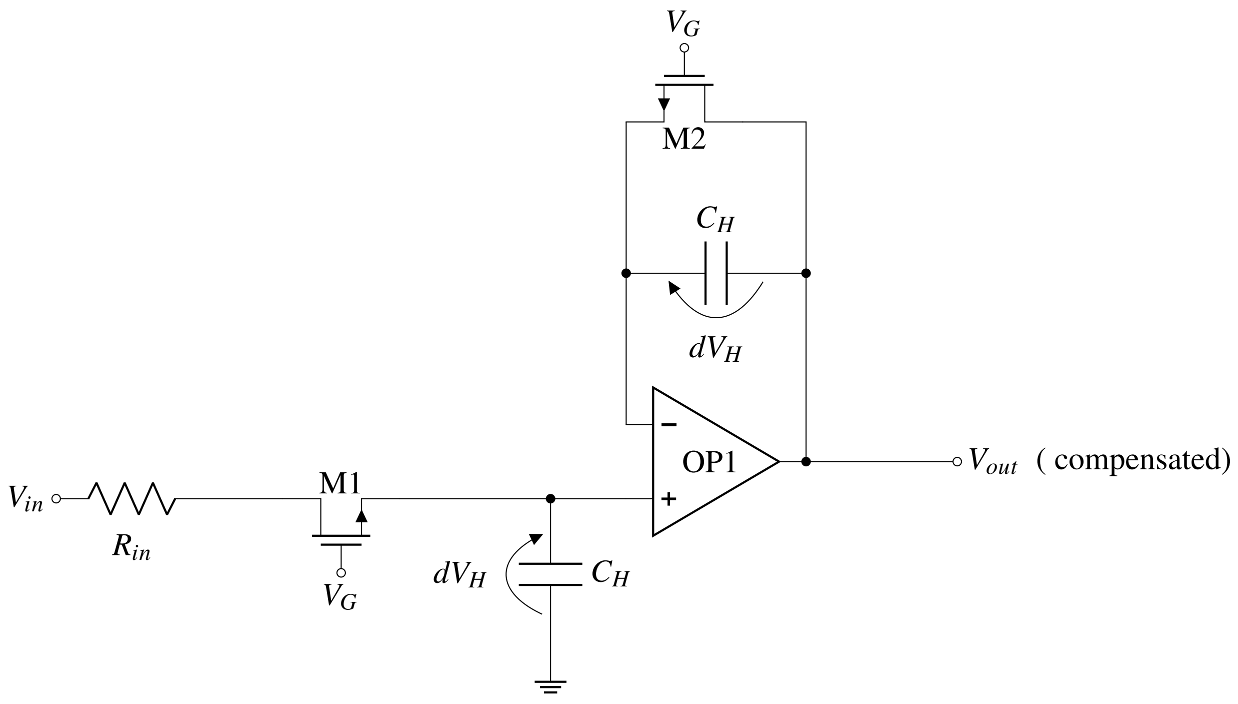


Figure 10.6: Sample and hold with charge injection compensation.

10.2.6 Feedback S&H

Another idea for a S&H circuit is to use a feedback loop to quickly charge up the capacitor at the speed of a closed-loop pole, while opening the loop and thus retaining the stored voltage during the hold phase. Figure 10.7 shows an idea for a possible implementation with opamps, although many more implementations may exist, like figure 10.8, which uses an OTA to charge up the hold capacitance in an integrator.

Both these designs have the advantage of speeding up transfer due to the closed-loop poles of the system, which must be computed through the loop gain. The main difference lays in the voltage at the source of M1: the OTA implementation makes that node a virtual ground, thus making charge injection independent of the value stored into V_H, which makes it possible to calibrate it out of the conversion system. Note that both these designs achieve a gain of $1 + R_2/R_1$, making them suited to acquire weak signals and amplify them. Also, note that the virtual ground at the source of M1 in the second design makes it possible

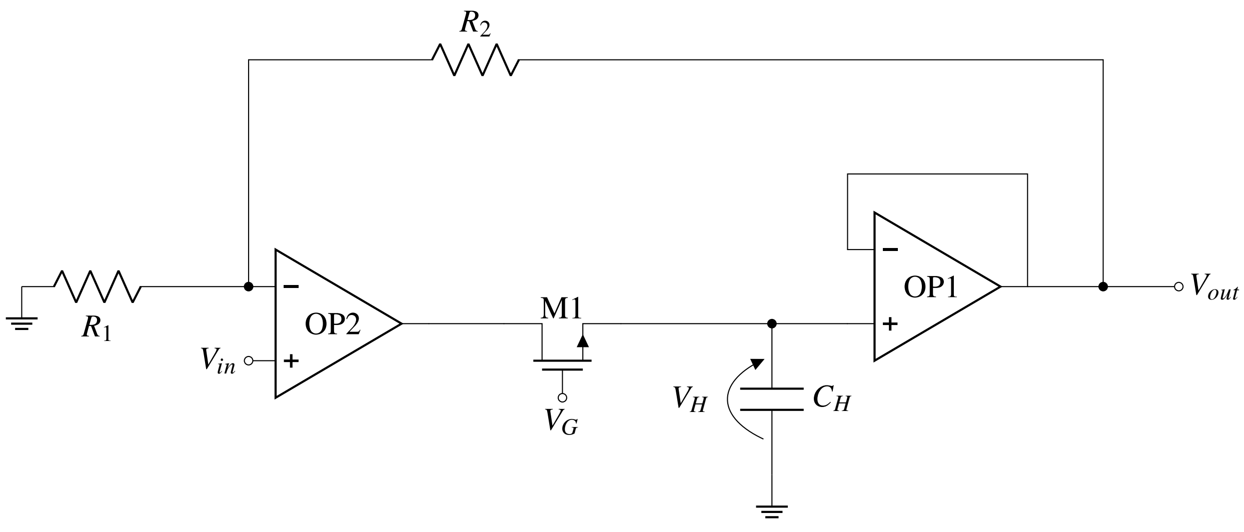


Figure 10.7: Feedback sample and hold stage using an opamp.

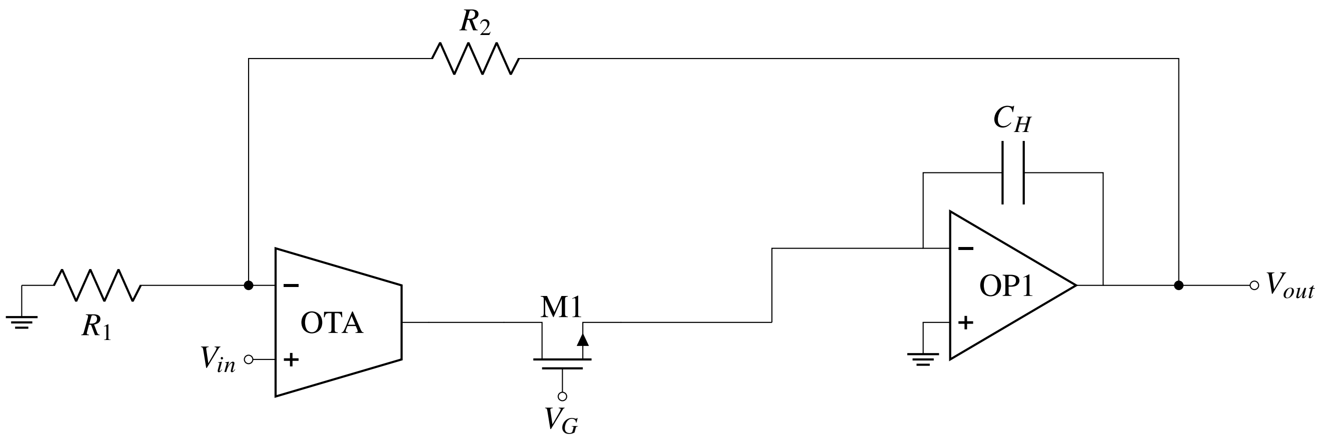


Figure 10.8: Feedback sample and hold stage using an ota and an integrator.

to move V_G by slightly more than V_T rather than by $V_T + V_{in}$, which is very convient as it allows it to be piloted directly by a standard logic gate, something not possible with the baseline S&H stage, where variations of V_G can easily be upwards of 9 V.
Both of these designs suffer from a significant effect of the offset of the input operational amplifier, which is amplified by the same gain as the signal and must be calibrated out. Furthermore, stability is always a concern in closed-loop systems and should investigated carefully to obtain a good phase margin. Many of the problems in the past years exams focused on addressing these issues, so I won't be delving too deeply into them here.

10.2.7 Double capacitor sample and hold

This sample and hold circuit is based on the idea of using a smaller hold capacitor to speed up sampling, while switching to a larger capacitor -that has been charged to the same value- to reduce the effects of all the errors previously seen during hold phase, which are reduced for larger C_H.

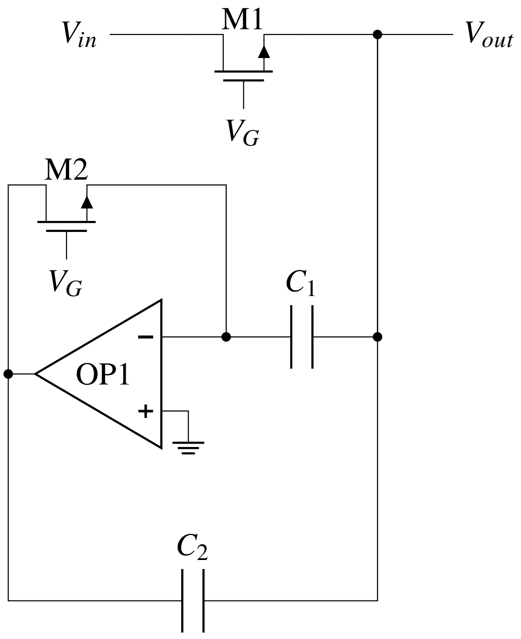


Figure 10.9: Double capacitor sample and hold

When the transistors are ON, the opamp has negative feedback. If we neglect the M2's R_{on}, C_1 and C_2 are in parallel, and act as a $C_1 + C_2$ capacitor that charge through M1's R_{on}. When the transistors are opened, on the other hand, the opamp's feedback acts to increase the equivalent capacitance. To understand this point, note that placing a test generator at the V_{out} leads to zero current flowing through C_1. Furthermore, the test current only flows into C_2, leading to a voltage drop $\frac{i_T}{sC_2}$ across it. Since the opamp's output is at A_0V_-, we may write the following equation by equating the voltage drop across C_2 with $V_{out\ opamp} - V_t$:

$$\frac{i_T}{sC_2} = V_t - A_0V_t = V_t(1 + A_0)$$

Note that since there is no current flowing into C_2, the voltage at the test node is equal to that driving the opamp. We get an input impedance, which is the ratio between V_t and i_t, equal to

$$\frac{V_t}{i_t} = \frac{1}{sC_2(1 + A_0)}$$

which is the same complex impedance seen across a capacitor that is as large as $C_2(1 + A_0)$. That's very large, and makes this stage almost immune to the issues seen above. Still, this is only true as long as the opamp doesn't saturate, which happens within a few μs. To realize this, just compute the expression for V_{out} when a current I_{out} is being sunk from the stage. As all the voltage is sunk by C_2, this will lead to a ramp rate of I_{out}/C_2 across the capacitor. It only takes a few microseconds for the voltage across C_2 to be as large as the power supply, and at the point the stage stops working, as the opamp cannot change its output to try to keep the ≈ constant voltage at the output node.

Chapter 11

MUX, DIGPOT, DAC and ADC

11.1 The analog multiplexer (MUX)

A multiplexer is a device used to reroute signals in an electronically controlled manner. This means connecting one terminal, usually seen as the output, to a multitude of input terminals. Doing this for analog signals requires using transistors as switches to connect and disconnect signals.

The circuit in figure 11.1 shows an example of how a pair of MOS transistors controlled by a digital signal CTR can be used as a switch, allowing to connect and disconnect the two input and output terminals. Using two transistors compared to one allows for larger signal dynamics and for charge injection compensation, as the charge needed to turn off one transistor comes from the other transistor in the pair. When the pmos is turned off, for instance, the positive holes from its channel recombine with the electrons coming from the channel of the nmos, thus leading to almost no net charge flowing into or out of the system. This is useful for sample and hold stages, where this charge leads to the problem of charge injection seen in the previous chapter.

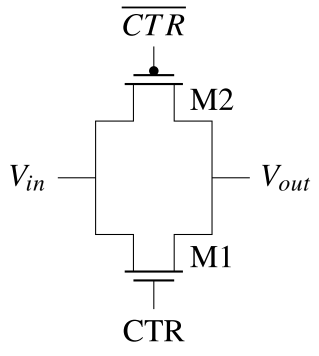


Figure 11.1: CMOS switch.

The full multiplexer just uses a lot of these switches in parallel, with appropriate control logic needed to decode the input signal, which is usually a binary number, into the specific ones and zeros turning on the switch related to that number and turning off all the other ones.

Please note that all the transistors will have a resistance R_{ON}, when active, that is not zero, and a resistance R_{off}, when turned off, that is not infinite. There is usually also a leakage current I_{off} affecting all the switches. These and all other important parameters are found in the datasheets of the components, and can be used to calculate the error they introduce at the output using a simple circuit model. In general these problems become worse as the number of channels increases, making the design of large analog multiplexers a complex affair.

One of the primary uses of these devices is found in multi-channel sample and hold stages. These systems use analog multiplexers to sample signals from multiple sources under digital control. Of course, the sampling rate of each source is reduced compared to what is possible by using a single channel, as the ADC's time will be split across the various inputs. This means that these systems offer limited bandwidth and are thus more widely used for applications where bandwidth isn't important, like temperature or pressure sensors. If a larger bandwidth is needed for some of the channels but not for all of them, a sampling scheme where those channels that require a higher bandwidth are sampled more often can be devised. For instance, let's consider a multiplexer with four channels, $C_0 \ldots C_3$. If the system is sampled as $C_0, C_1, C_2, C_3, \ldots$ each channel gets an equal sampling rate that is $1/4$ the sampling rate of the converter. Instead, if the channels are sampled as $C_0, C_1, C_0, C_2, C_0, C_3 \ldots$ the C_0 channel gets sampled more often. Every six sampling cycles, three are given to C_0, while one is given to each of $C_1 \ldots C_3$. This means that C_0 will get a sampling rate that is half that of the ADC, meaning a larger possible bandwidth. This, of course, sacrifices the sampling rates of all the other inputs, which are now $1/6$ of that of the ADC.

There are two types of analog MUX available: ”make before break” and ”break before make”. The first, when switching from one channel to another, first closes the second switch, and then opens the first. This creates a brief shortcut between two of the channels, but prevents the signal path from being closed entirely during the transition, which may be desirable, for instance, within a feedback loop. There, a brief interruption in the feedback path may cause the system to saturate.
The other kind -break before make- first opens the initial switch, and then closes the second switch. This makes it desirable to avoid shorts between different inputs, but breaks the signal path completely for a brief transition time.

11.2 The digital potentiometer (DIGPOT)

The digital potentiometer was developed as a device showing a variable resistor between three input terminals. It is made by a resistive partition connected to an analog multiplexer and an output buffer. Note how this design, unlike real resistors, is unidirectional, meaning that the buffer reads the selected voltage and forces it at the C terminal, but a voltage applied there (which would short the buffer and likely damage it), doesn't affect the other two terminals. This means that these devices are often used with terminals A and B connected to voltage sources, allowing a selectable voltage to be shown at node C. Also note how the switches are in fact made by transistors as in figure 11.1, requiring a decode logic to select the correct one.

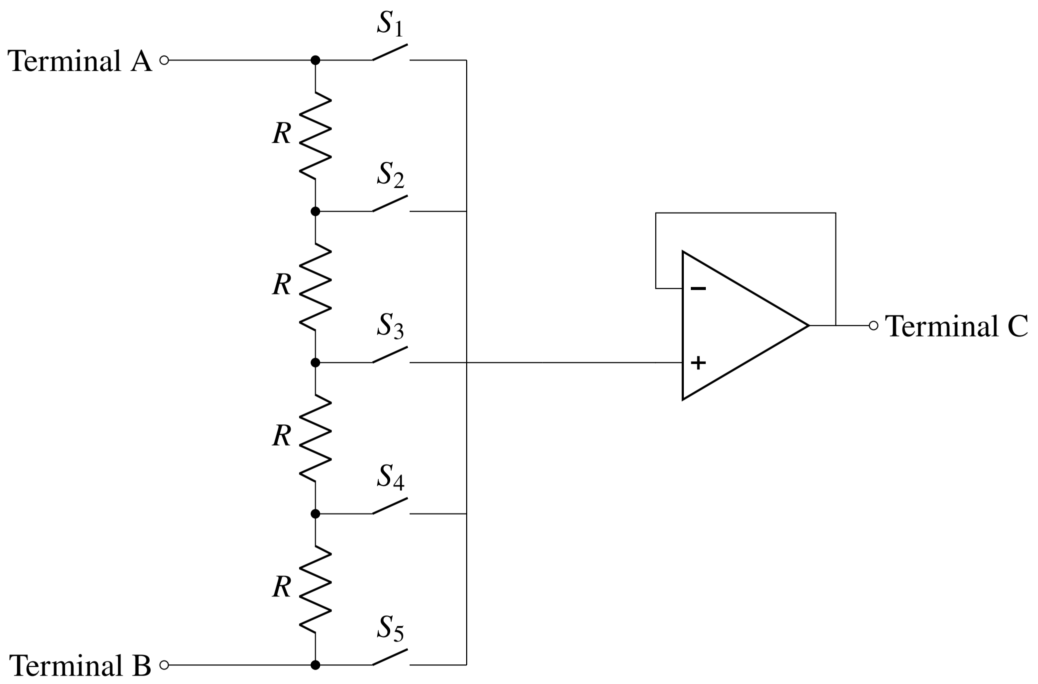


Figure 11.2: Digital potentiometer

11.3 Digital to analog converters (DAC)

A digital to analog converter is a device which receives a number D_{in} encoded as binary as its input, and outputs an analog number $V_{out} = V_{ref}\frac{D_{in}}{2^n}$, where n is the number of bits. It is used every time a digital signal, like a song in a smartphone, needs to be converted to an analog signal, for instance to control a stereo.

11.3.1 Voltage scaling DAC

The simplest idea to design a DAC involves the digital potentiometer shown in figure 11.2. When the A terminal is connected to V_{ref} and the B terminal to ground, the input digital number selects the appropriate voltage on the resistor ladder, which is then put at the output. This is a very simple and effective design for a small number of bits, but it is not scalable: the number of resistors must be doubled for every extra bit that is needed.

11.3.2 Weighted-R DAC

The weighted-R DAC attempts to fix the issue of the voltage scaling DAC by starting from the simple observation that the each 1 in the input digital word corresponds to an output voltage that depends on its position within the binary number. In general, a 1 at a position i in a binary number corresponds, when converted to decimal numbers, to a value $< 2^i$. This means that for every 1 in the input word, we have to increase the output voltage by $V_{ref}\frac{2^i}{2^n}$, while for every 0 nothing should be added. This can be practically implemented using a voltage adder circuit with an opamp[1], whose inputs are either V_{ref} or ground depending on the value of that bit, and whose weights scale as 2^n depending on the position of the bit. The least significant bit (D_0) should be scaled by a weight that is $1/2^n$, the second bit (D_1) should be scaled by $1/2^{n-1}$, and so on until the most significant bit (D_{n-1}), which should be scaled by a factor $1/2$. Since the gain of each input of an inverting adder is $-\frac{R_f}{R_i}$, the most significant bit will require a resistor $R_{n-1} = 2R_f$, the bit before that will require $R_{n-1} = 4R_f$, all the way to the least significant bit, which will require $R_0 = 2^n R_f$. Also note that the implementation shown in figure 11.3 uses an inverting adder, meaning that due to the inverting gain of the adder each input, when its switch is closed, should be connected to $-V_{ref}$. although the number of resistors in this design is equal to the number

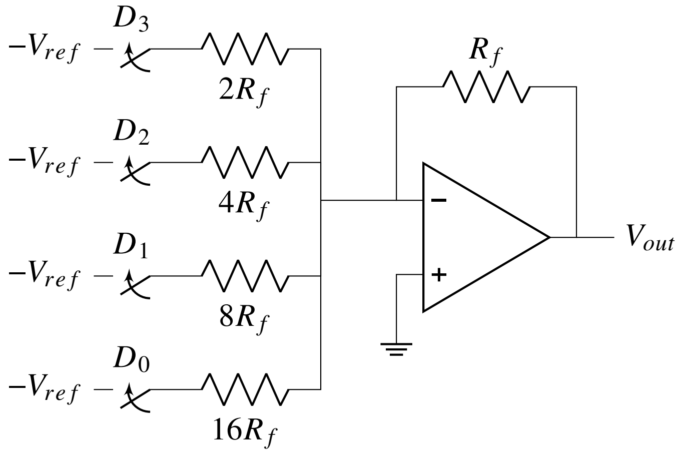


Figure 11.3: Binary Weighted-Resistor Digital-to-Analog Converter (4-bit example). The resistors scale by powers of 2 to assign binary weights to the input bits.

of bits plus one (for the feedback resistor), their values quickly grow out of hand. The least significant bit requires a resistor with a very large value, which must be implemented, once

[1]see figure 3.4

the minimal technological width is being used, by a very long resistor[2]. As the area taken thus scales exponentially with the number of bits, this solution is also not scalable and should only be used for a small number of bits.

11.3.3 R-2R ladder DAC

The idea of the R-2R ladder DAC is to use a very peculiar resistive net, known as the R-2R ladder, to select the voltages, already scaled as powers of two, that should be added together to form the output voltage. An opamp adder is used as before, along with switches selecting which voltages should be added or not depending on if the input bit is a 0 or 1. As seen before, the least significant bit adds a voltage that is $V_{ref}/2^n$, the second bit adds a voltage $V_{ref}/2^{n-1}$, and the most significant bit adds a voltage that is $V_{ref}/2$. The idea of the R-2R network is to take the previous voltage and apply to it another resistive partition to half it. This is cleverly achieved by the network in figure 11.4, where the resistance seen at each intersection between a resistor R and $2R$, towards ground, is always $2R//(R+R) = R$. This implements the voltage partition described above with just 2 resistors per bit, making this a highly scalable and very clever solution This still requires the fabrication of many

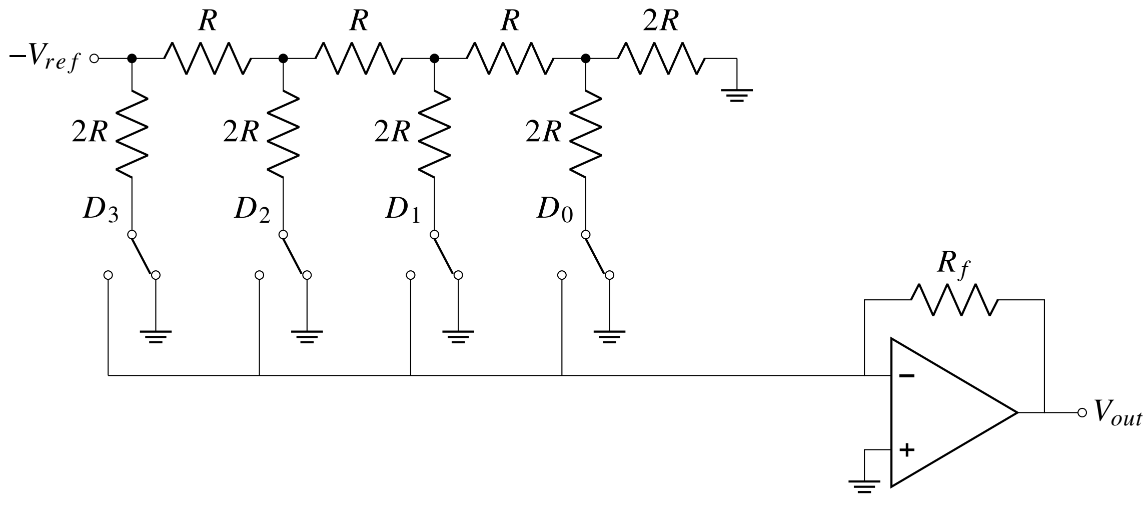


Figure 11.4: R-2R ladder DAC

precise resistors, which might prove expensive in integrated circuits. Solutions based on capacitors can be used instead.

11.3.4 Binary weighted capacitor DAC

The idea of this DAC is to rely on capacitive partitions rather than resistive ones, thus avoiding the fabrication of resistors. As capacitors can be made on a very small area, this is a cheaper solution than those involving resistors. This DAC has a set of capacitors in

[2]Remember that a resistor follows the formula $R = \rho\frac{L}{W \cdot t}$. Selecting t_{min}, ρ_{max} and W_{min}, L is the only free parameter.

parallel, which are connected at the top rail to the output node and to a switch to ground (out of which no current should flow), and at the bottom either at ground or V_{ref}. At first, they are all connected to ground and discharged, meaning that $Q_{tot} = 0$. Once they have all been discharged, the switch connecting the top rail to ground is closed. As this node is insulated, the total charge inside this node must not change from the initial condition. As we modulate the bottom switches, connecting some of the capacitors to ground and others to the reference voltage (depending on if the respective bit is zero or one), the voltages across the capacitors will vary to maintain the total charge at the output node, since for every capacitor it must be that $Q_i = C_i V_i$. If we call $C_{tot} = \sum_i^n C_i$ and C_{active} the capacitance of the capacitors connected to V_{ref}, and $C_{passive}$ the capacitance of the capacitors connected to ground, we can write the charges stored in the two groups as: $Q_{active} = C_{active}(V_{out} - V_{ref})$ and $Q_{passive} = C_{passive}(V_{out} - 0)$. This means that the total charge stored in the capacitors is $Q_{tot} = Q_{passive} + Q_{active} = C_{active}(V_{out} - V_{ref}) + C_{passive}(V_{out} - 0)$, and, since the reset condition discharged all of them at the beginning, it must now be that $Q_{tot} = 0 = C_{active}(V_{out} - V_{ref}) + C_{passive}(V_{out} - 0)$. Solving for V_{out} we find the potential at which the output node must go to preserve the charge in the system, which is

$$V_{out} = V_{ref} \frac{C_{active}}{C_{active} + C_{passive}} = V_{ref} \frac{C_{active}}{C_{total}}$$

Based on this result, we can see that the output voltage will change depending on which capacitors are connected to V_{ref}. As seen above, the output voltage will need to be the sum of several contributions from the active bits, each scaled by its own power of 2. The most significant bit will contribute to $V_{ref}/2$, the bit at its right will contribute with $V_{ref}/4$, and so on until the least significant bit, which will contribute $V_{ref}/2^n$. For this to happen, if $C_{tot} = C \cdot 2^n$, it must be that the capacitor associated to the most significant bit must be $C_n = C \cdot 2^{n-1}$. The one at its right must provide a factor $1/4$, and thus be $C_{n-1} = C \cdot 2^{n-2}$, and so on until the least significant bit, whose capacitor must account for a small $1/2^n$ factor, and thus be $C_0 = C$. Note that in order for all the capacitors to add up to $C \cdot 2^n$, as assumed above, it is needed to add another capacitor always connected to ground at the beginning of the parallel. Also, note that the output node cannot draw any current: it is a floating node which must be carefully read by a voltage buffer with no bias current and minimal parasitic resistance. The complete design is shown in figure 11.5. Note that these capacitors are often arranged in concentric circles on the IC, in a common centroid structure aimed at minimizing mismatches. Since this class doesn't cover IC layout, this isn't an important detail to remember or understand.

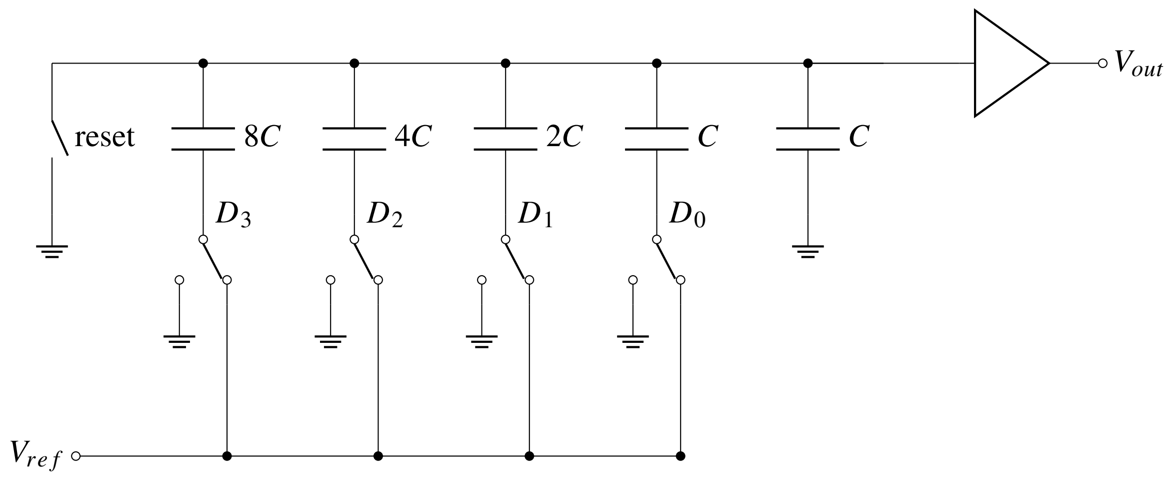


Figure 11.5: Binary-weighted capacitor DAC.

11.4 Analog to digital converter (ADC)

Analog to digital converters are circuits that take an input analog signal and produce as output a binary number describing the analog value at their input as $D_{out} \approx 2^n \frac{V_{in}}{V_{ref}}$, where the approx is due to the discrete nature of binary number, which cannot represent a continuous amount of input analog values. In fact, an ADC static input-output curve will be a staircase with continuous input values and discrete output values. There are many kinds of ADC as each one has distinct advantages and disadvantages. The designer must therefore know all of them and be able to select the most appropriate one to meet his target specifications.

11.4.1 Flash ADC

A flash ADC is the fastest type of ADC. It consists of a resistive divider between ground and V_{ref}, aimed at representing a number of voltages between the two, and a set of comparators connected to the input signal. The comparators turn on for all values below their threshold, thus enabling an encoding logic that follows to determine the correct output word. The comparators' output, which is all ones up to that level, is called ”thermometric code”, and it has has many bits as the number of levels. This means that this design requires a very large number of comparators, equal to 2^n, making not very scalable to large number of bits. The conversion time, however, is in the order of the settling time of the converters at open loop, which, depending on their poles, can be very fast, in the ns range. Figure 11.6 shows the analog part of the design for a two bit converter. As a final remark, note that a simple differential stage with mirror is more than enough to implement those comparators, and that a full opamp is not needed, although the symbol may suggest that. This is because the comparators are used to drive cmos logic gates, which are basically a capacitive load.

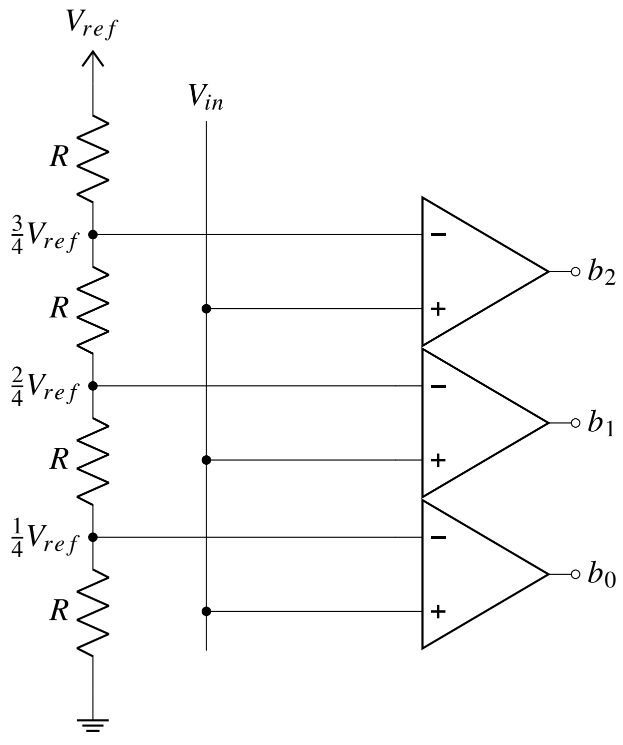


Figure 11.6: Flash ADC. Note that more logic is needed to go from the the $b_0 \dots b_2$ thermometric code to the output of the ADC, which is just two bits.

11.4.2 Staircase ADC

A staircase ADC uses a binary counter that is connected to a DAC and a comparator. As the ADC's clock switches, the counter increases its value, creating a staircase ramp out of the DAC that grows, from zero, of 1 LSB at each tick of the clock. This value is continuously compared to the output value by an open loop opamp, that outputs a one once $V_{DAC} > V_{in}$. At that point, the conversion has ended and the number stored in the counter is the converted value, which can be read by an external digital system. The worst case conversion time is the time required by the input staircase to reach the highest possible input, which is V_{ref}. This takes as many as 2^n clock cycles, meaning that the worst-case conversion time is $T_{clk} \cdot 2^n$, which can be long, in the order of μs. Still, this design only requires one comparator and is thus rather cheap to build, making it suited for all sensors requiring low frequency acquisition, like temperature transducers. Figure 11.7 shows the prototypical block diagram. Note the S-R latch, which is used to turn on the counting at the beginning of conversion and turn it off at the end of conversion, by taking its $\bar{Q}$ output and ANDing it the external clock via an and gate. The start of conversion signal is at the R terminal of the latch, thus bringing $\bar{Q}$ to 1 when the start of conversion signal goes high. This allows the clock to work and the counter to function properly. Note that for this to start at zero every time, the counter must be reset by the start of conversion signal.

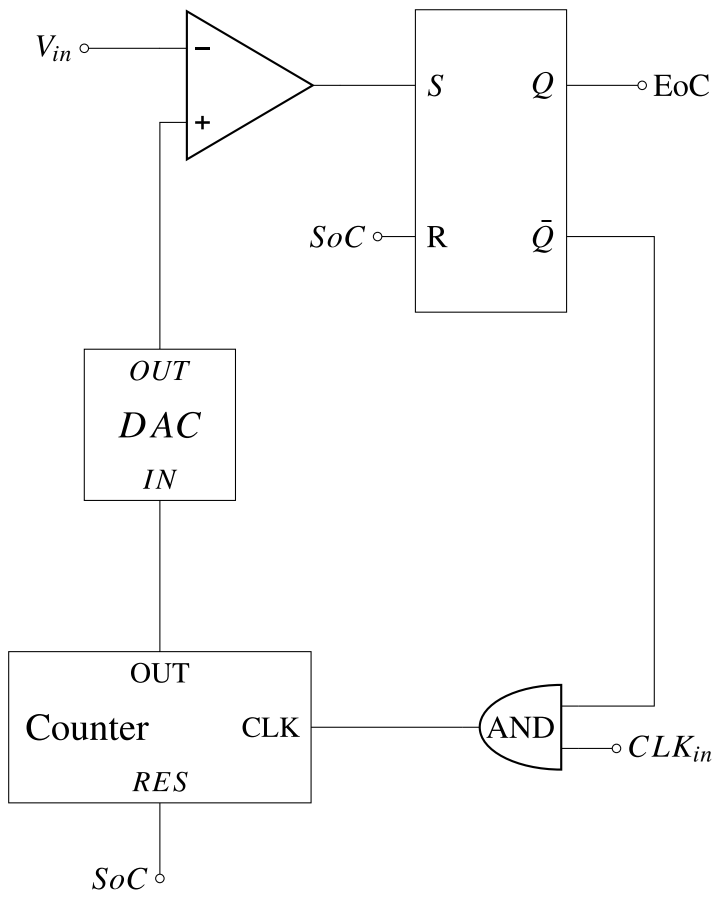


Figure 11.7: Staircase ADC block diagram.

11.4.3 Tracking ADC

The tracking ADC is built on a similar concept as the staircase adc shown in figure 11.7. The idea to improve its performance is not to reset the counter at every new conversion, but instead to count up or down (depending if the output of the comparator is above or below the input) until the two signals cross and the output of the comparator flips. This means that usually only a few clock cycles are needed for a conversion. Of course, when the circuit is first powered on, the system will have to first acquire the signal, thus responding with a slow ramp like the staircase ADC. Only once the signal has been converted once, the ADC can work as intented. Note that this tracking behavior poses a limit to the slope of the input signal, which cannot be greater than the amount of travel of the DAC's output in the same time in order to avoid losing track of the signal. As the slope of that staircase is about $\frac{LSB}{T_{clk}} = \frac{V_{ref}}{2^n \cdot T_{clk}}$, this sets a limit, for instance, to the maximum frequency of an input sinusoidal signal. Of course, if this system is used with a S&H stage this isn't a problem. As a last remark, note that this design can be very fast, being only limited by the comparator and DAC's response, which can be in the hunderds of ns range.

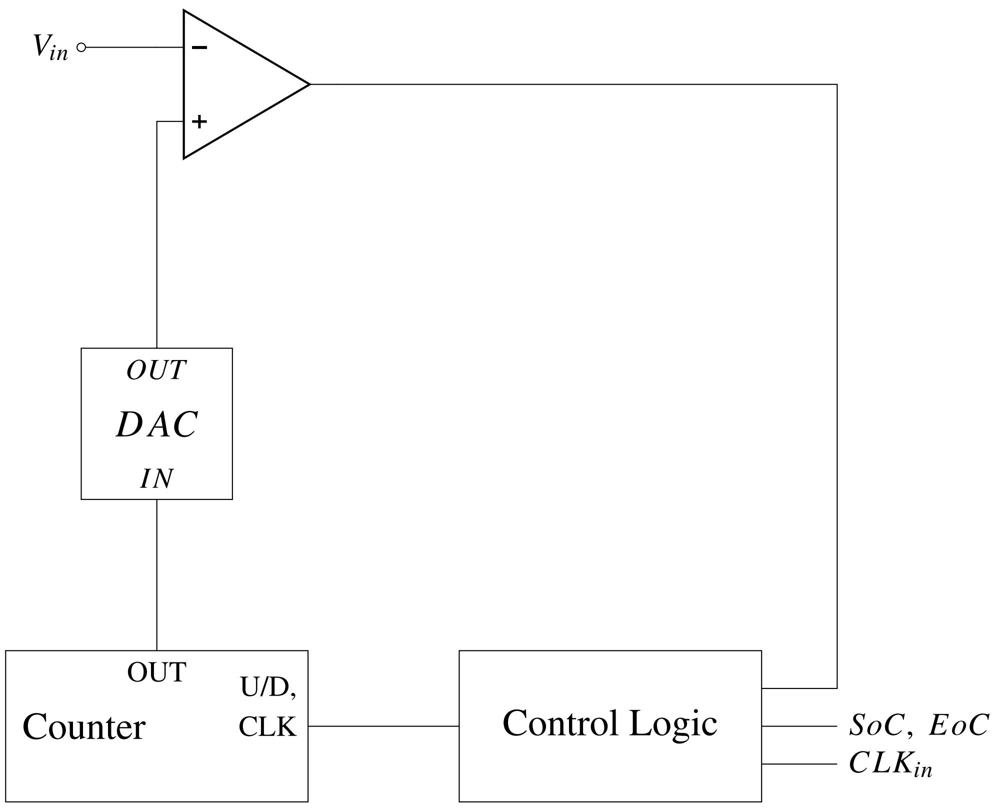


Figure 11.8: Tracking ADC block diagram.

11.4.4 Single and double slope ADC

The idea of a single slope ADC is very similar to that of a staircase ADC, but the use of DAC is avoided by relying instead on an analog ramp generator, made by injecting a constant current into a capacitor. As the slope at which the capacitor charges is I_C/C, it is possible to know, based on how long it took for the ramp to reach V_{in}, what was the V_{in} value. In practice this is achieved by running a binary counter at such a rate as that in T_{clk} the analog ramp has increased by exactly 1 LSB. Usually, the clock frequency is fixed and known, and the capacitor and current generator are sized to achieve this. Of course, little mismatches in the capacitor's value directly influence the output, causing a non negligible error. Still, this is an extremely cheap way to obtain an ADC with a large number of bits, although the statistical error due to mismatches of I_C/C may very well reduce the effective resolution of the instrument

To ensure that the capacitor has discharged completely once the conversion has ended, another comparator must be employed, and the capacitor must be discharged below the baseline reference value of the signal, which is ground. This means that this design, which is shown in figure 11.9, requires dual power supplies. It's also a slow design, very much and even more than the staircase ADC, as it requires the counter to go all the way to V_{in} following the I_C/C slope, which means that anywhere from ms to hunders of μs may be needed.

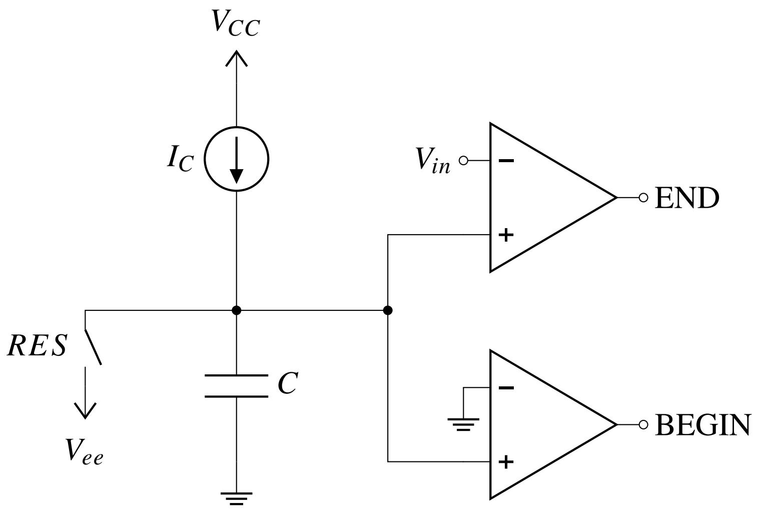


Figure 11.9: Single slope ADC. Analog components only.

As mentioned above, this stage suffers from errors due to the tolerances in the capacitor and current source. A way to avoid this is the so called "double slope" ADC, whose structure is shown in figure 11.10. The basic idea of this stage is to run the integration of current twice: once it is run with a current equal to $-V_{in}/R$ for exactly the time needed for 2^n clock pulses, starting with the capacitor discharged, and then it is run with a current that is V_{ref}/R, until the output hits 0 V again. As the current flowing into the capacitor imposes the relation $\frac{dV_C}{dt} = \frac{I_C}{C} = -\frac{V_{in}}{RC}$, the first ramp is a straight line with equation $V_C(t) = -\frac{V_{in}}{RC}t$, reaching after 2^n clock pulses a value of $V_C(t = T_{clk} \cdot 2^n) = -\frac{V_{in}}{RC}T_{clk} \cdot 2^n$. After this, the integrator switches to the input, making the slope $\frac{dV_C}{dt} = \frac{I_C}{C} = \frac{V_{ref}}{RC}$, and thus obtaining $V_C(t') = \frac{V_{ref}}{RC}t' + V_C(t_0)$[3], with $V_C(0) = \frac{V_{in}}{RC}T_{ref} \cdot 2^n$ as computed above. If we ask ourselves for what value of $t' = T_k$ we get $V_C(t) = 0$, we find: $\frac{V_{ref}}{RC}T_k = \frac{V_{in}}{RC}T_{clk} \cdot 2^n \longrightarrow T_k = T_{clk}\frac{V_{in}}{V_{ref}}2^n$, meaning that exactly $\frac{V_{in}}{V_{ref}}2^n$ clock pulses have elapsed since the beginning of the integration of V_{ref}, and exactly $\frac{V_{in}}{V_{ref}}2^n + 2^n$ since the beginning of the conversion. Thus, by simply counting the number of clock pulses in the second stage of the conversion, the input value can be determined in a manner that is independent on R and C, whose values cancel out. In this regard, the first counting can be regarded as a sort of calibration run of the instrument, used to modify the starting point of the second run as to compensate for tolerances. Also, note that once the conversion is complete it is needed to short the capacitor to reset the stage, hence a switch has been put across it in the design.

Another important property of double slope ADCs is their immunity to noise at certain frequencies, which are rejected by the integration, as it acts as a sort of "averaging out" of the input signal. To understand this point, let's assume that the input is affected by a sinusoidal disturbance of amplitude A_0 and radial frequency ω, hence $V_{in}(t) = V_0 + A_0 \sin(\omega t)$. Only the first stage of integration is affected, as its slope will be $\frac{I_C}{C} = \frac{V_{in}(t)}{RC}$, and therefore

[3] $t' = t - T_{clk} \cdot 2^n = t - t_0$

$V_C(t) = V_C(t_0) + \int_{t_0}^{t} \frac{V_{in}(t)}{RC} dt = \int_0^{t'} \frac{V_0 + A_0 \sin(\omega t)}{RC} dt = \frac{V_0}{RC} t' + \frac{A_0(1-\cos(\omega t'))}{RC\omega}$. We can see that if $t' = (2\pi)/\omega$, the third term, which is due to the disturbance, cancels out. This means that the duration of the first integration -and thus the clock period, as it will last for $t' = T_{clk} \cdot 2^n$- the noise's effect is rejected. This is commonly used as a constraint to select a proper T_{clk} for rejecting the 50Hz noise commonly found due to AC coupling with the electrical power lines. Also note that all frequencies multiple of that specific ω, due to the periodicity of cos are also rejected, and that frequencies close to that, while not entirely rejected, are also attenuated.

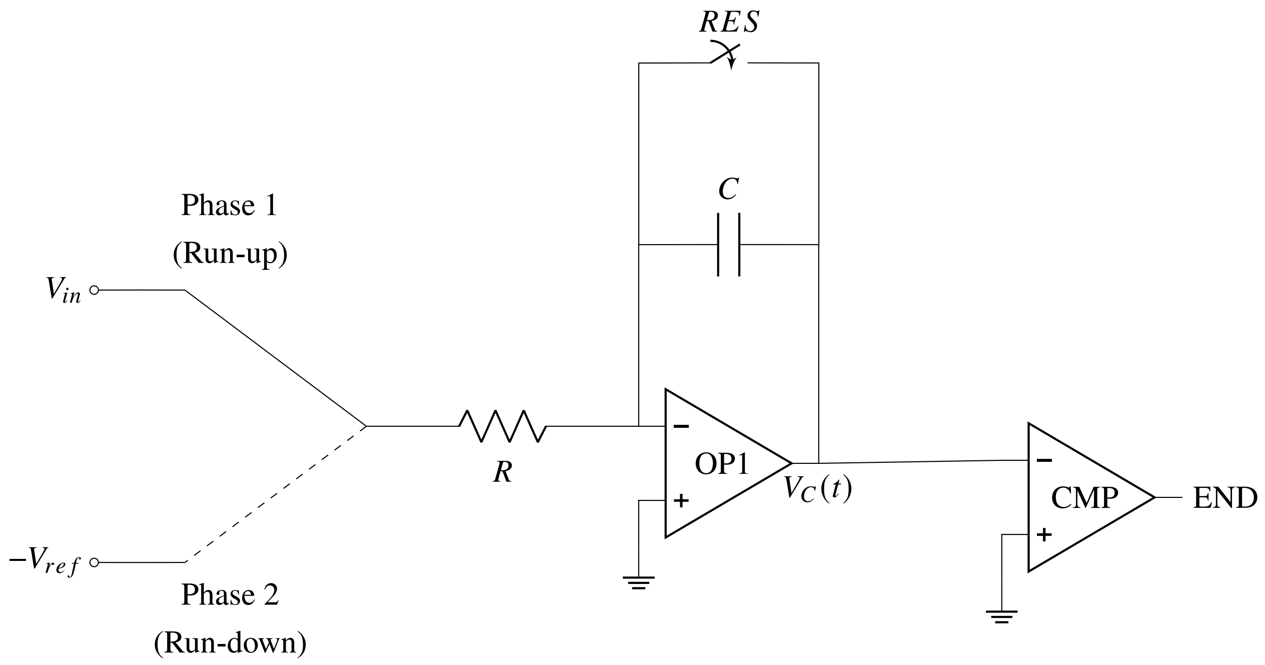


Figure 11.10: Double slope ADC. Analog components only.

Note that the $-V_{ref}$ value is obtained from an inverting configuration stage with gain -1. Also, figure 11.11 has been addded to clarify the response of the stage.

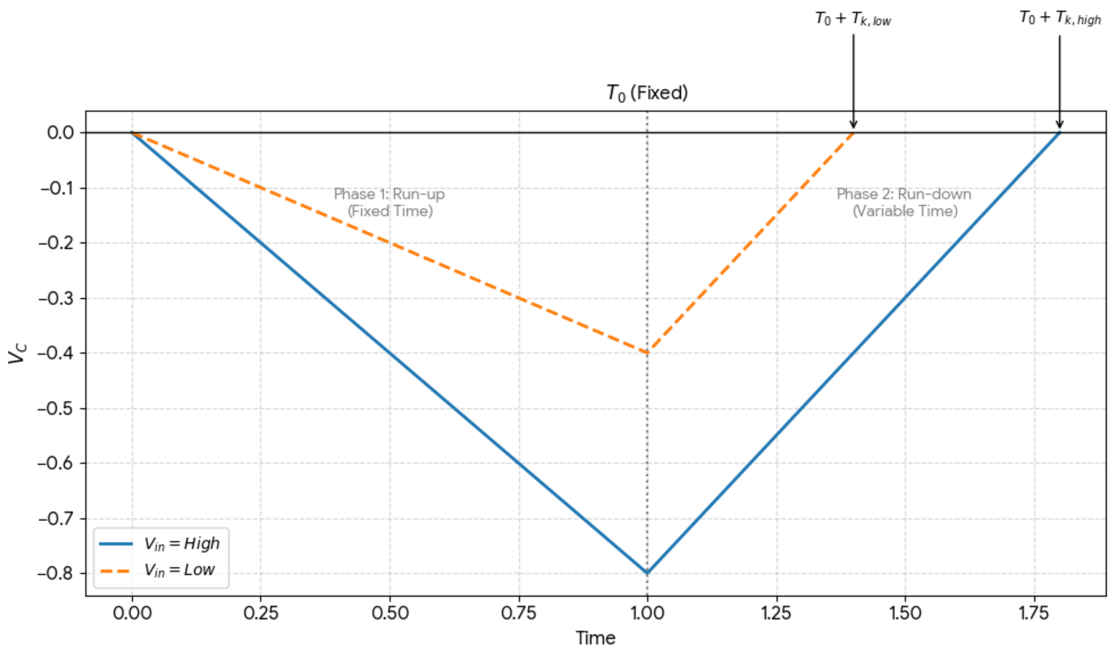


Figure 11.11: The first slope is for integrating V_{in}, the second for integrating V_{ref}

11.4.5 Successive-approximation ADC

The successive approximation ADC is a further improvement to the idea of the staircase ADC. Instead of comparing the input signal with all possible values through a staircase, we might find the right value faster by bisection. This involves comparing the input value, at first, with half V_{ref}, figure out if it's in the top half or the bottom half, and then comparing it with either $V_{ref}/4$ or $3V_{ref}/4$, and so on until all bits have been established. In fact, since the way binary numbers work is precisely by powers of two, the algorithm we have just seen corresponds to asking ourself if the most significant bit should be 1 or 0, then if the bit at its right should be 1 or 0, and so on until the least significant bits. This ensure n bit precision in $n + 1$ clock cycles, where one last step in needed at the beginning of the sequence for the S&H stage before the ADC to acquire the input value. This makes these ADC very fast and allows them to obtain a large number of bits without compromising speed or the number of components, as DAC with many bits can be constructed easily as seen above. The architecture is very simple as far as the analog components are concerned, as it really only needs a DAC and a comparator. It is shown in figure 11.12.

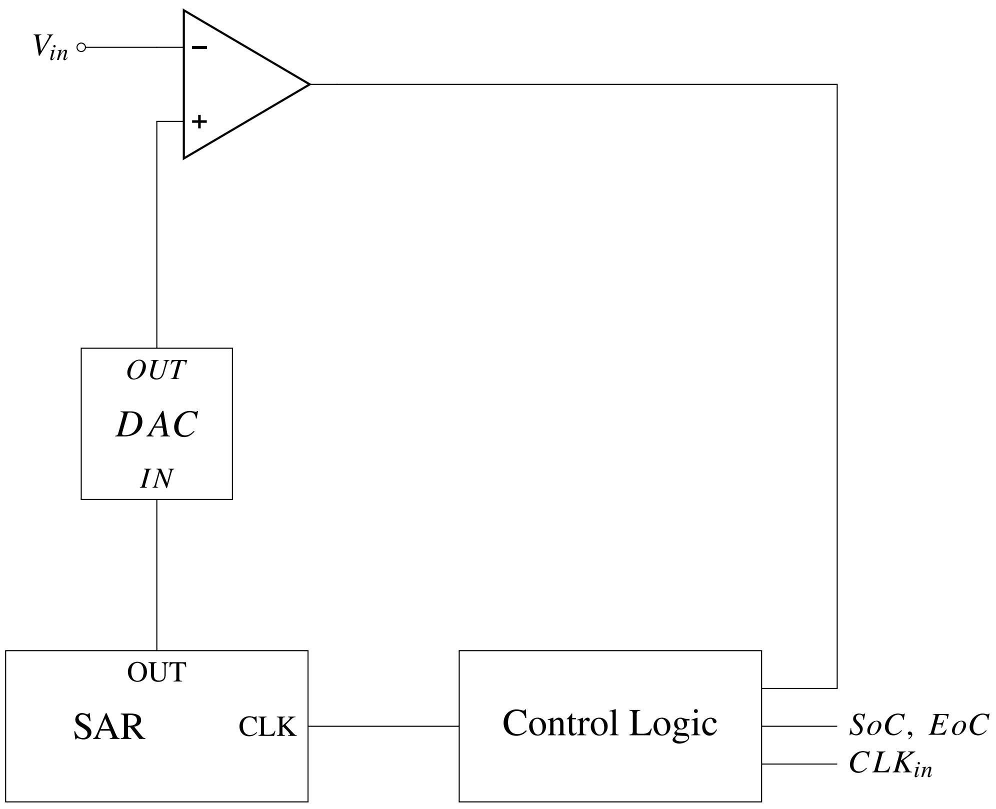


Figure 11.12: SAR ADC block diagram. Note that SAR stands for successive approximation register.

11.4.6 Sub-ranging ADC

The idea of a sub ranging ADC is to use consecutively two flash ADC for the most significan bits and for the least significant bits, thus saving a large number of comparators, but requiring a two step conversion process. For this to work, we should first determine the most significant bits by comparing the input to a voltage partition, and then use a finer voltage partition network for the least significant bits. In practice, this is efficiently achieved by the architecture in figure **??**, represented for a two bit adc. As seen below, this design first determines the level at which the input is by means of the comparators to the right, with all the switches open. Then, depending on that level, it must close the row of switches between the last comparator to be active and the last one to be inactive. This way the interval between these comparators is further split into sub-ranges by the resistive divider that is created there, which the lower comparators can use to determine the fine position of the input. As with the flash ADC, this requires translating from the thermometric code used by the comparators (all ones up to the level of the input) to a binary number representing it, which can be achieved by an appropriate digital lookup table.

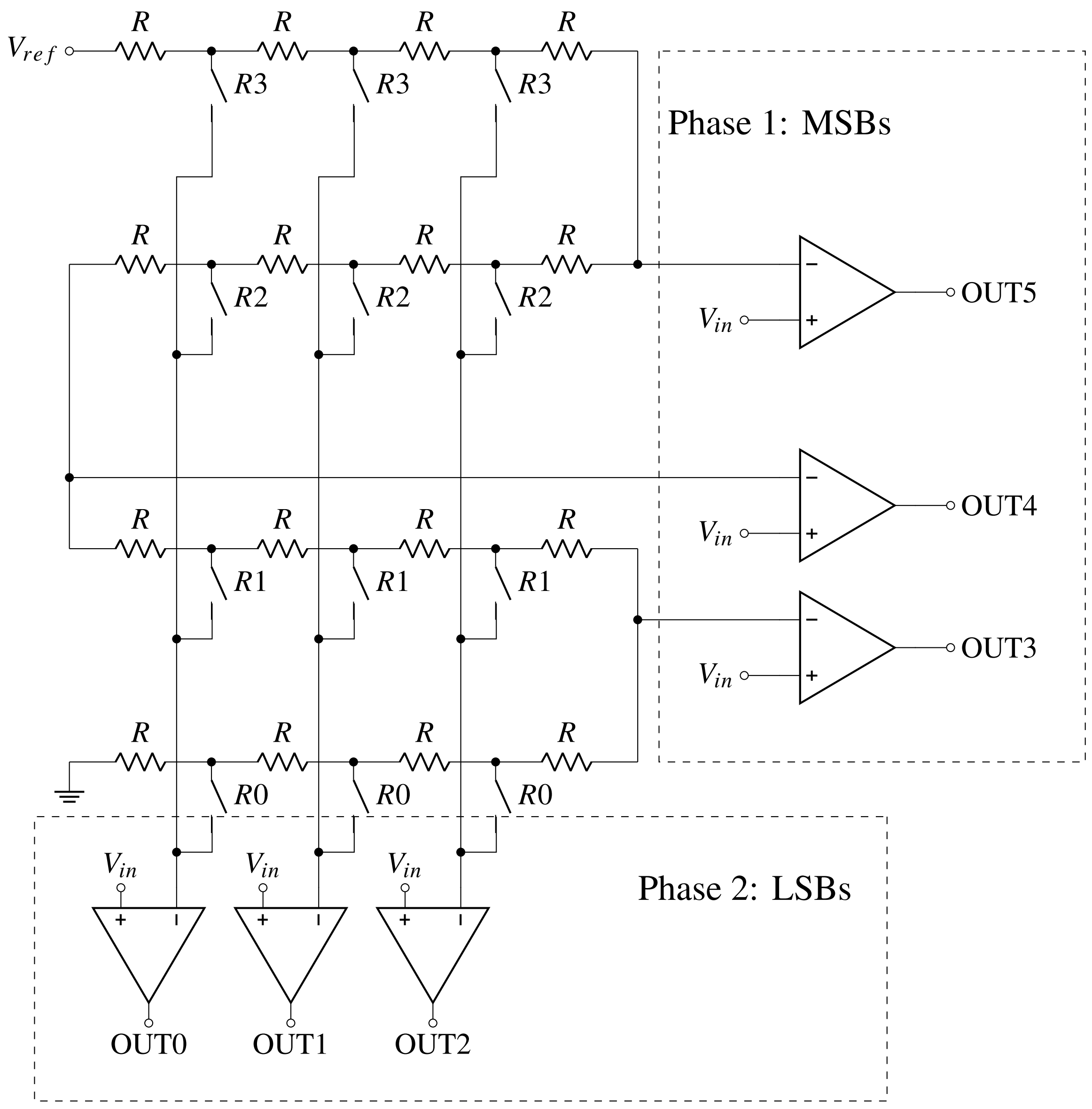


Figure 11.13: Subranging ADC. Analog parts only.

This design requires only $2^{n/2} - 1$ comparators for the MSB, and as many for the LSB, thus scaling better than the flash ADC to large numbers of bits. Note that the number of resistors needed is still 2^n, but, as they can be made of any size, this poses little constrain on the area occupied by the resistive partition. As a last remark, note that altough the switches connect the LSB comparators to the right row of resistor in order to compare the input to those values, the resistive partition always works in the same manner, with all subdivisions of V_{ref} always being present, even when they are not being used by any comparator.

11.4.7 Interpolation flash ADC

The idea of the interpolation ADC is to use less comparators than the traditional flash ADC, but to have them operate in their linear region. This way, the missing outputs can be reconstructed by means of a resistive partition between the existing comparators. Figure 11.14 shows this architecture.

As CMP1 and CMP2 both operate in their linear region, their outputs will be $V_1 = A_0(V_{in} - \frac{5}{6}V_{ref})$ and $V_2 = A_0(V_{in} - \frac{1}{6}V_{ref})$. (Note that the voltages at their - terminals are determined by the asymmetrical input voltage partition.) As a result, the resistive divider between their outputs will see a potential difference across it of $V_2 - V_1 = A_0 \frac{4}{6} V_{ref}$, meaning that the intermediate voltages will be: $V_3 = A_0(V_{in} - \frac{4}{6}V_{ref})$, $V_4 = A_0(V_{in} - \frac{3}{6}V_{ref})$, $V_5 = A_0(V_{in} - \frac{2}{6}V_{ref})$. These are exactly the voltages we'd have out of linear comparators connected to the initial resistive partition at the left (like in the flash ADC). Depending on if those voltages are positive or negative, the value of V_{out} with respect to V_{ref} in thermometric code can then be established, and from that more digital logic downstream can perform the conversion to a binary number. This means that digital buffers with a threshold 0 need to be connected to $V_1, \dots, V_5$ to regenerate those analog voltage levels into reliable digital ones, which must be as close as possible to either power supply. These buffers can then output to digital latches (like SR latches) that sample the signal in the digital domain, thus avoiding a sample at hold stage all together.

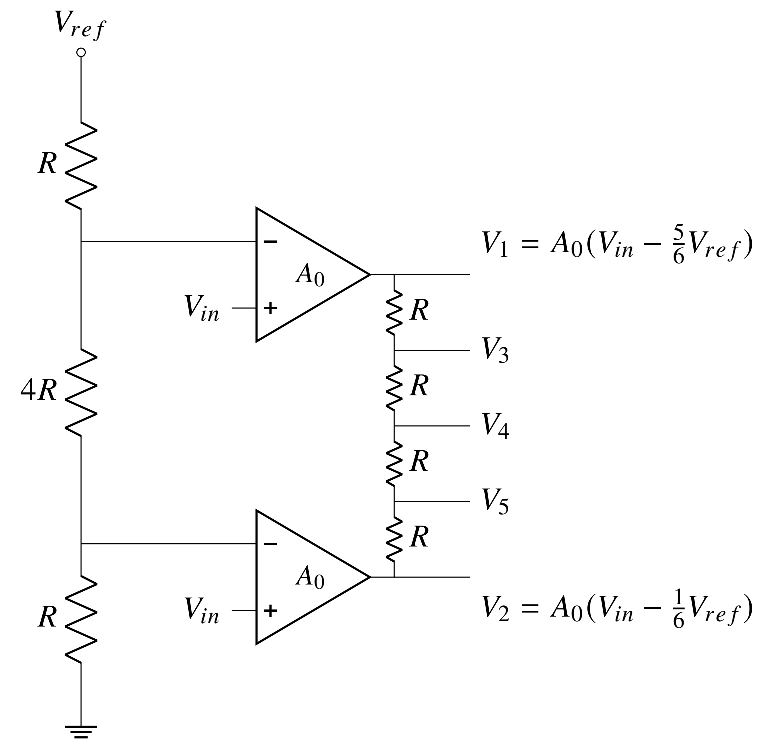

V_{ref}
R
$4R$
R
V_{in}
V_{in}
A_0
A_0
−
+
−
+
R
R
R
R
$V_1 = A_0(V_{in} - \frac{5}{6}V_{ref})$
V_3
V_4
V_5
$V_2 = A_0(V_{in} - \frac{1}{6}V_{ref})$

Figure 11.14: Example of an interpolation ADC. Analog components only.

Note that the behavior described above assumes that the comparators are operating linearly. This is only true as long as their outputs don't saturate to the power supply. For this reason it's important to choose a low A_0 and an appropriate power supply, which must be at least $\approx \pm V_{ref} + V_{in}$ to confortably fit the outputs without getting close to saturation.

11.4.8 Folding flash ADC

The folding flash ADC was developed from the sub-ranging ADC seen above. That design used two separate $n/2$ flash ADCs for the most significant and the least significant bits, but it required a two-step acquisition process, making it always slower than a flash ADC. To improve on this, a good idea would be to use a folding circuit that non-lineary processes the input, folding it into subranges by subtracting from it the MSB value, but before it is computed by the other flash ADC. This can be achieved by a nonlinear circuit using rectifying amplifiers, or even in a microelectronics stage. Let's assume that a folding stage has been designed with a static transfer characteristic like the one in figure 11.15. This stage can be connected before the LSB flash ADC to obtain the finer details, while the MSB flash ADC can work directly with the input to roughly figure out its value. Figure 11.16 shows the block diagram for the complete design. Note that as all the components in this architecture are feed-forward solid state systems, the total propagation delay will be in the tens of ns range, making this a very fast architecture. As a final remark, note that if the LSB flash ADC and the MSB flash adc don't have the same number of bits, the architecture can still be employed with a different folding scheme.

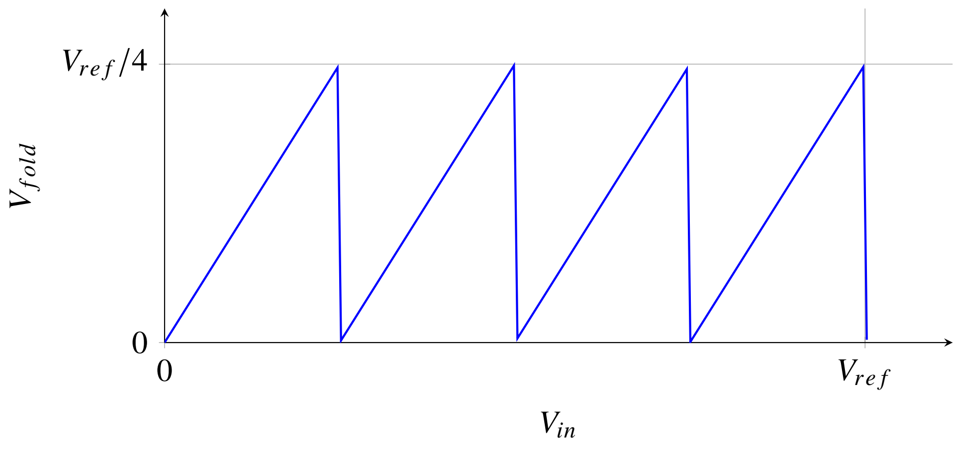


Figure 11.15: Sawtooth folding transfer function. $y = x - \text{floor}(x)$. An input value sees subtracted from it the last bit covered by the MSB ADC.

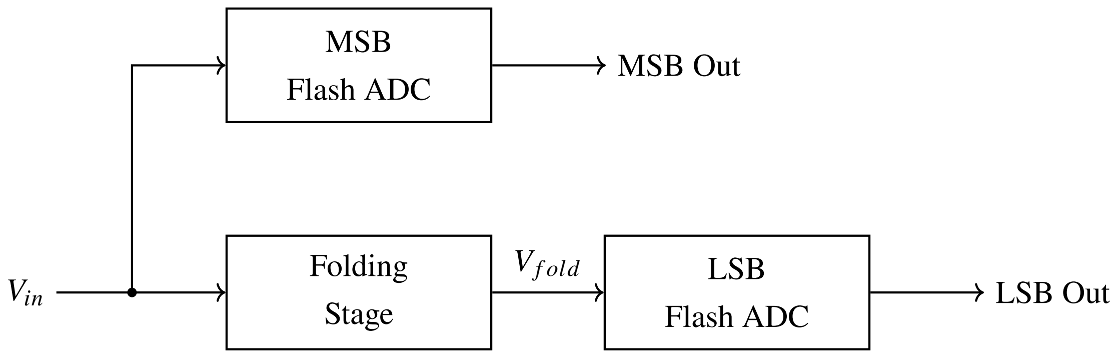


Figure 11.16: Simple block diagram of a Folding ADC architecture.

11.4.9 Multi-step ADC

The idea of a multi-step ADC is, once again, to use two (or more) flash ADC with less bits to save on comparators. Instead of relying on a complex non-linear folding stage like the folding ADC, we can instead go on and actually subtract from the input the analog value corresponding to the MSB using an opamp subtractor stage. Upon doing this, we can then send the result to a further flash ADC for the LSB. Figure 11.17 shows an example of this design for an ADC with two stages, also called *Half-flash ADC*.

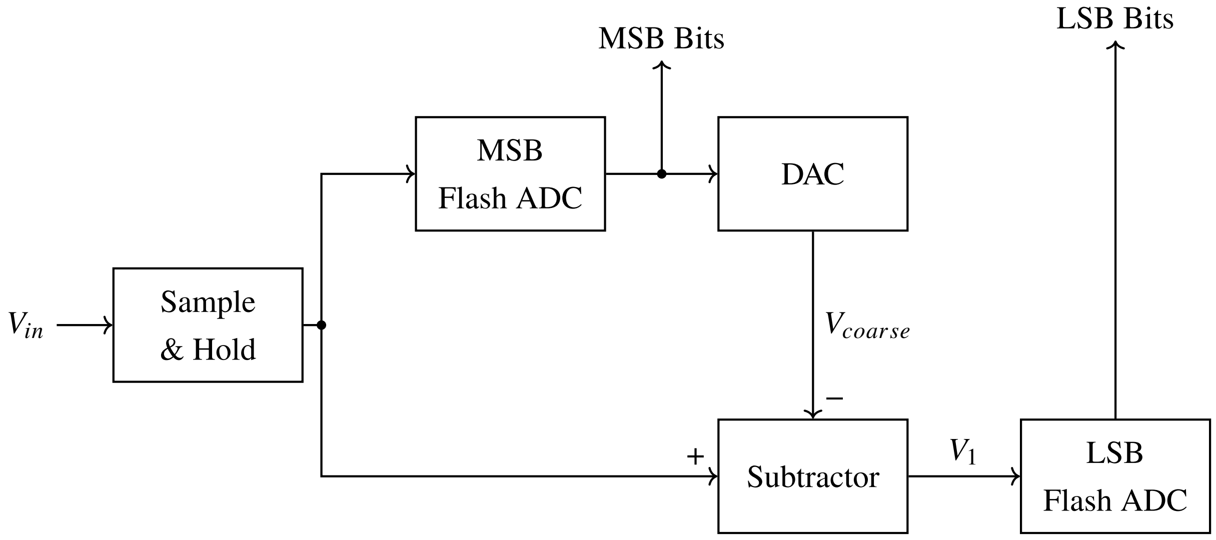


Figure 11.17: Simple block diagram of a two-step (Half-Flash) ADC architecture.

The input is first fed into a sample and hold stage. Then it is sent to both a MSB coarse flash ADC and to the subtractor. The output of the first ADC is then fed into a DAC (more on this detail later), and to the subtractor, whose output is now $V_1 = V_{in} - V_{coarse}$, which is then fed into the LSB flash ADC. Note that this architecture is entirely feed-forward, with a maximum frequency that is limited by the delay of all the stages in series.

ADC+DAC

Altough the block diagram in figure 11.17 would imply that an ADC and a DAC should be used in series, there is a better implementation that allows to save significant area. The basic idea is to recycle the reference resistors for the flash ADC for use in the DAC. This can be achieved by simply using an appropriate combinational logic to select which one of the values already present on the resistive divider of the flash ADC should be put out, by means of a MOS switch. The value we should choose is the one between the last active comparator and the first active comparator. This means that the logic function needed to control the switches should output a 1 (to close the switch) when its inputs are 01 or 10, and a zero for 00 and 11. This is just the XOR gate, which should then be connected between two subsequent comparators. In practice this is like mixing together a voltage scaling DAC and a flash ADC.

11.4.10 Pipelined ADC

A pipelined ADC is very similar to a multi-step ADC, with the addition of a sample and hold circuit in between the stages. This enables them to work independently: the first stage can start working on a new sample while the second one is working on the last sample, and so on. This scales extremely well with a large number of stages and enables this ADC to break the tradeoff between stages and speed that existed in the multi-step ADC. This enables pipelined ADC to operate at very high speeds and with very high precision, making them essential in modern high bandwidth analog instruments like oscilloscopes. Figure 11.18 shows the design for just two stages, but more could be cascaded if a larger number of bits is needed.

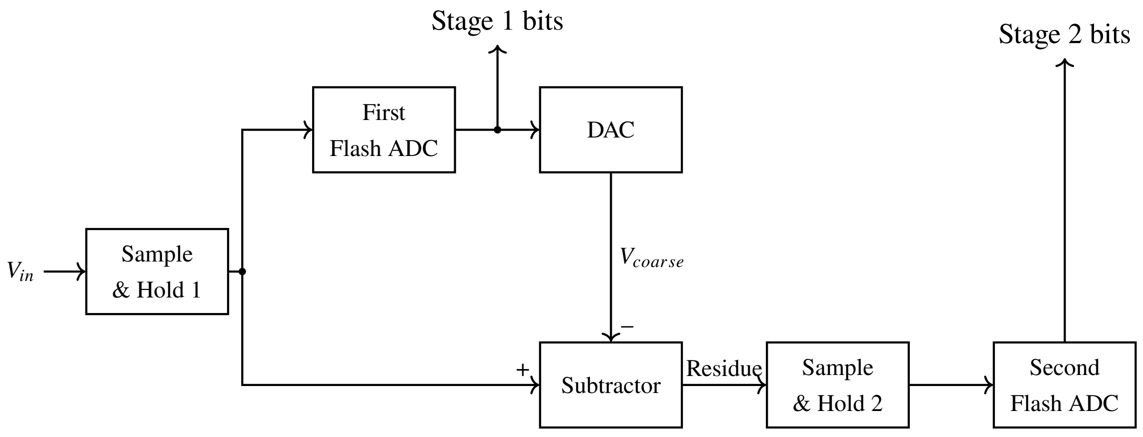


Figure 11.18: Simple, two stages, pipelined ADC.

Note that an architecture like this requires a digital pipeline in parallel with the analog one, to keep track of the previous bits coming from past stages. At any given time, the output of each stage will refer to different samples and should therefore not be mixed.
This design achives very high sampling speeds, which are only limited by the slowest of the stages. Latency, instead, is worse, as for a sample to be fully converted it needs to go through all the stages. Usually this isn't a problem, as at the sampling rates at which these devices are operated it is still possible to get the final sample within microseconds of it entering the pipeline.

Chapter 12

Oversampling and Sigma-Delta

12.1 Oversampling

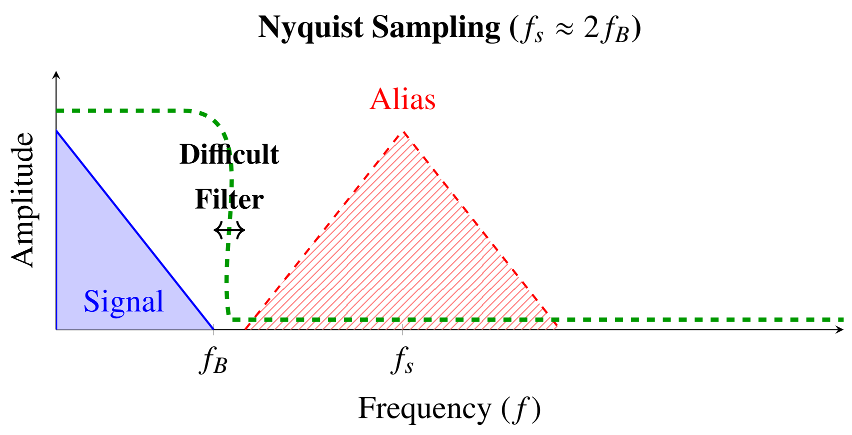
Oversampling refers to sampling an analog signal with bandwidth B at frequencies much greater than 2B. This has two primary advantages:

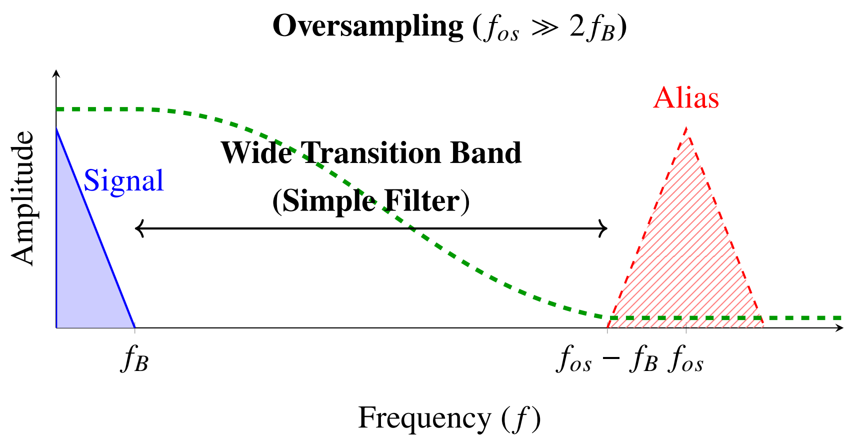
- reduction of the discretization noise
- relaxed requiremets on the anti-aliasing filter
- improvement of the resolution through

Let's analyze all of them, starting from

12.1.1 Relaxed requirements for the anti aliasing filter

As mentioned in the chapter on sampling, in order to prevent the input from having frequency components above the Nyquist limit of $f_s/2$, where f_s is the sampling frequency, a filter is added to remove them. As illustrated below, sampling at frequencies above the Nyquisit limit, due to the periodicity of the Fourier transform of the sampled signal, which is f_s periodic, leads to much more room for the anti aliasing filter to remove those components, thus reducing the order of the needed filter and making it easier to implement.





12.1.2 Noise reduction

Oversampling reduces the *noise floor* due to quantization noise. Basically, when a signal is quantized, an error is introduced due to approximating the continuos analog value with discrete analog ones. If the signal is distributed evenly across the ADC's input dynamic, it's possible to prove that the average squared value of this error is $\sigma^2 = \frac{LSB}{12}$, where LSB is the resolution of the ADC. As the error has zero average, its mean squared value (power) equals its variance. This error, being random in nature, can be modelled as white noise, which is constant in the frequency domain. This means that the quantization noise's power, σ^2, can be though of as a constant noise in the spectrum 0 to $f_s/2$, with power spectral density equal to $\frac{\sigma^2}{f_s/2}$. As we increase f_s well above the Shannon limit, the power spectral density due to quantization noise decreases, as the same noise power is spread over a larger bandwidth. Ultimately, this is like having the quantization noise (in term of power spectral density, or noise floor), of an ADC with a larger number of bits and therefore higher resolution. If we sample at a frequency that is larger than the Nyquist limit by a factor K[1], we get a noise power spectral density that is $\frac{\sigma^2}{f_s/2} = \frac{\sigma^2}{KB}$, which is a factor K lower than what would be obtained while sampling at the Shannon limit (K=1).

[1]K is the oversampling factor, equal to $\frac{f_s}{2B}$, where B is the unilateral signal bandwidth. Note that for $K = 1$ we are at the Shannon limit.

12.1.3 Extra bits

As far as the signal to noise ratio is concerned, nothing has changed: the ratio between the signal and the total area of the noise's power spectral density has not changed, as we have the same noise power spread over a larger banwidth. Here, however, something clever can be done: if the signal, once acquired, is filtered by a digital low pass filter with bandwidth B or so, the extra white noise beyond that bandwidth can be eliminated. In practise, oversampling has allowed us to decouple, in frequency, quantization noise and signal, allowing for a removal of the first. This means that, after filtering the signal, the total remaining power is just that within the bandwidth $\pm B$, which is $\frac{\sigma^2}{KB} \cdot B = \frac{\sigma^2}{K} = \frac{LSB}{12K}$. Compare that to the average squared value of the quantization noise which we had before applying the filter ($\frac{LSB}{12}$): it has been reduced by a factor K. This means that the signal to noise ratio has also been *increased* by a factor K! In dB, this corresponds to ad increase of the SNR (signal noise ratio) by a factor $10\log_{10}(K)$, which means that for K=2 we get an increase of $\approx 3dB$. As the SNR due to the quantization error can be written as $SNR_{id} \approx 6n + 1.76$, it means that for every quadrupling of K the signal to noise ratio due to quantization error diminushes by $10\log_{10}(4) \approx 6$ dB, which is as much as adding a new bit to the ADC!

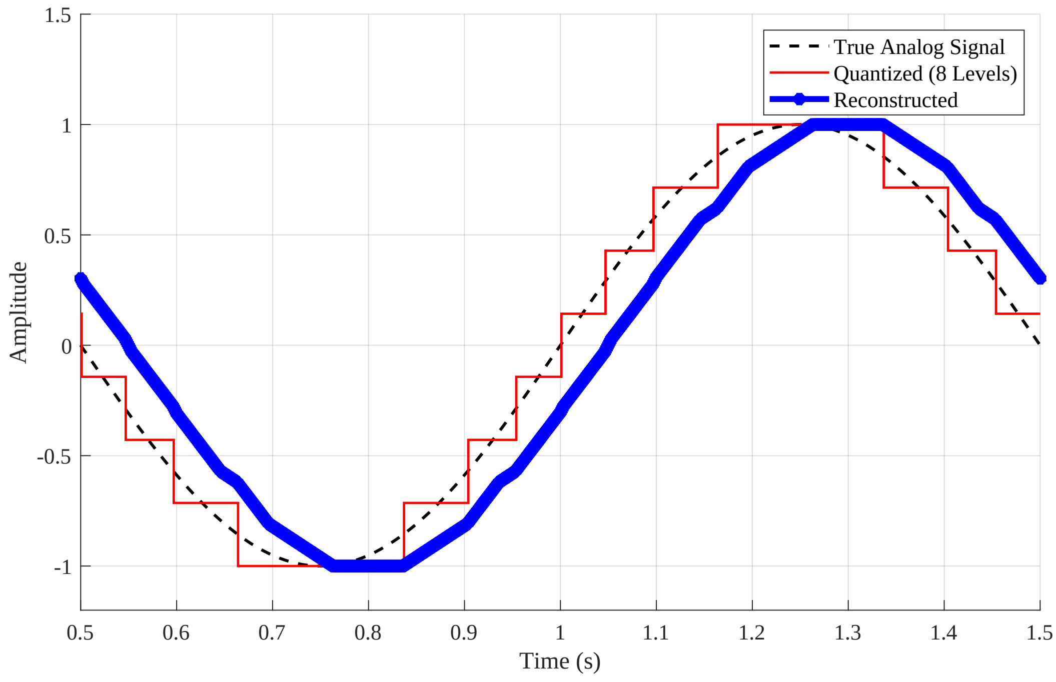


Figure 12.1: Effect of a moving average filter on a 8 bit oversampled sinusoid.

To understand what this filtering corresponds to in time domain, let's consider the case of a simple moving average filter, which simply makes the convolution between the sampled signal's vector and a new vector of length m and value $1/m$ for every component. This filter removes the quantization noise by smoothing out the rough samples around the points at which they transition. Figure 12.1 shows this process in action. Note that for this to work as intended, many samples must be provided, as the moving average calculates the values in between (which is what reduces the quantization noise) by an appropriate weighted average of all the sourrunding values. Also, please note that this derivation assumes that the input signal is uniformly distributed on the input dynamics of the ADC. This is crucial for a scheme like this to work: if the input were to remain constant, for instance, the error due to quantization would also be constant and thus at 0Hz, and the moving averge wouldn't be able to remove it.

12.2 Sigma-Delta modulator

The sigma-delta modulator is a fast ADC built to take advantage of oversampling. To understand how it works, let's first consider the so called "delta modulator" shown in figure 12.2. This is similar to a tracking ADC, but its output is simply a single bit value telling us if the signal, compared to the previous sample, has increased or decreased. If this is run quickly enough, the signal may be reconstructed by integrating (summing, in discrete time) the output of the modulator. We can see, then, that by *oversampling* a signal, with a 1 bit ADC, we can achieve the precision of an ADC with more bits. As summing the output of the modulator allows to reconstruct the input signal, we may use an integrator in the feedback loop.

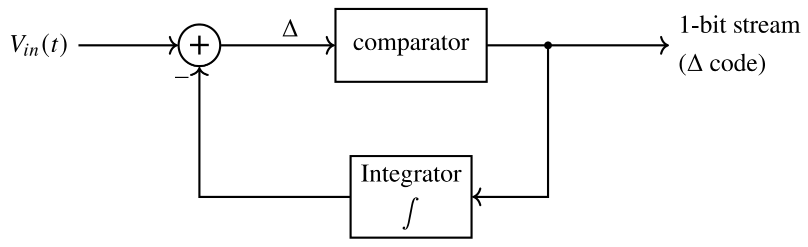


Figure 12.2: Delta Modulator block diagram.

The sigma-delta modulator has the structure seen in figure 12.3. It's born out of the idea of moving the integrator needed to reconstruct the output of the delta modulator into the modulator itself. Performing the required operations, it's possible to prove that the two are equivalent as far as the input → reconstructed signal relationship is concerned. Note that the system must feed a digital information into an analog adder, and then into an analog integrator, both of which are the same seen in chapter 3 of these notes.

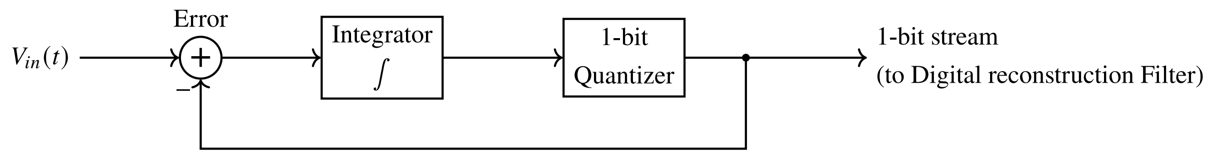


Figure 12.3: Block diagram of a simple sigma-delta modulator.

Just like any good system with feedback, this one attemps to cancel out the error at its input, but, being a discrete time system, it will do so on average. This means that the average value of the output will be the same as the value at the input, which is a nice way to think of what the system is doing in practise: trying to match the average value at the output with the input, in discrete time.

Noise shaping

The quantization noise of a sigma-delta modulator is very different than that of any other ADC studied so far. To study this point, the Z transform is needed, and that's beyond the scope of these notes. Suffice to say that the averaged squared value of the quantization noise is still $\frac{LSB^2}{12}$, but its shape in the frequency domain won't be constant. Instead, the noise power spectral density has $\alpha \sin(\pi f T_S)^2$ shape, meaning it's zero at the sides of the frequency spectrum, and larger at the center. This gives us, much like with traditional oversampling, an improvement in the quantization noise by increasing f_S. However, this improvement is much greater than before, due to the shape of the power spectral density function. This time, for every doubling of f_S, we get an increase in the SNR that we'd obtain by increasing the number of bits by ≈ 1.5 bits.

Chapter 13

Analog filters

Analog filters are physical implementations of transfer functions $|T(s)|$ whose poles and zeros allow for some processing of a signal. In these notes we'll analyze filters as abstract mathematical transfer functions first, and then we'll see some ideas on how they could be implemented in circuits.

13.1 First order filters

First order filters are filters with a single zero and a single real pole. Their transfer function is of the kind $T(s) = \frac{a_1+sa_0}{s+\omega_0}$. In chapter one and three of these notes their implementations has already been seen by means of simple RC networks and opamp stages like the integrator and derivator. In general they can be classified as either low-pass filters, with transfer functions of the kind with a single pole and no zero ($T(s) = \frac{a_1}{s+\omega_0}$), or high pass, with a zero in the origin ($T(s) = \frac{sa_0}{s+\omega_0}$). The Bode plots of these configurations are shown in figure 13.1, with a gain of 1 (0dB). They can be used to remove frequency components outside the pass region, but, as evident in the Bode plots, their gentle $\pm 20\frac{dB}{dec}$ slope doesn't allow for removal of frequencies that are very close to their pass regions.

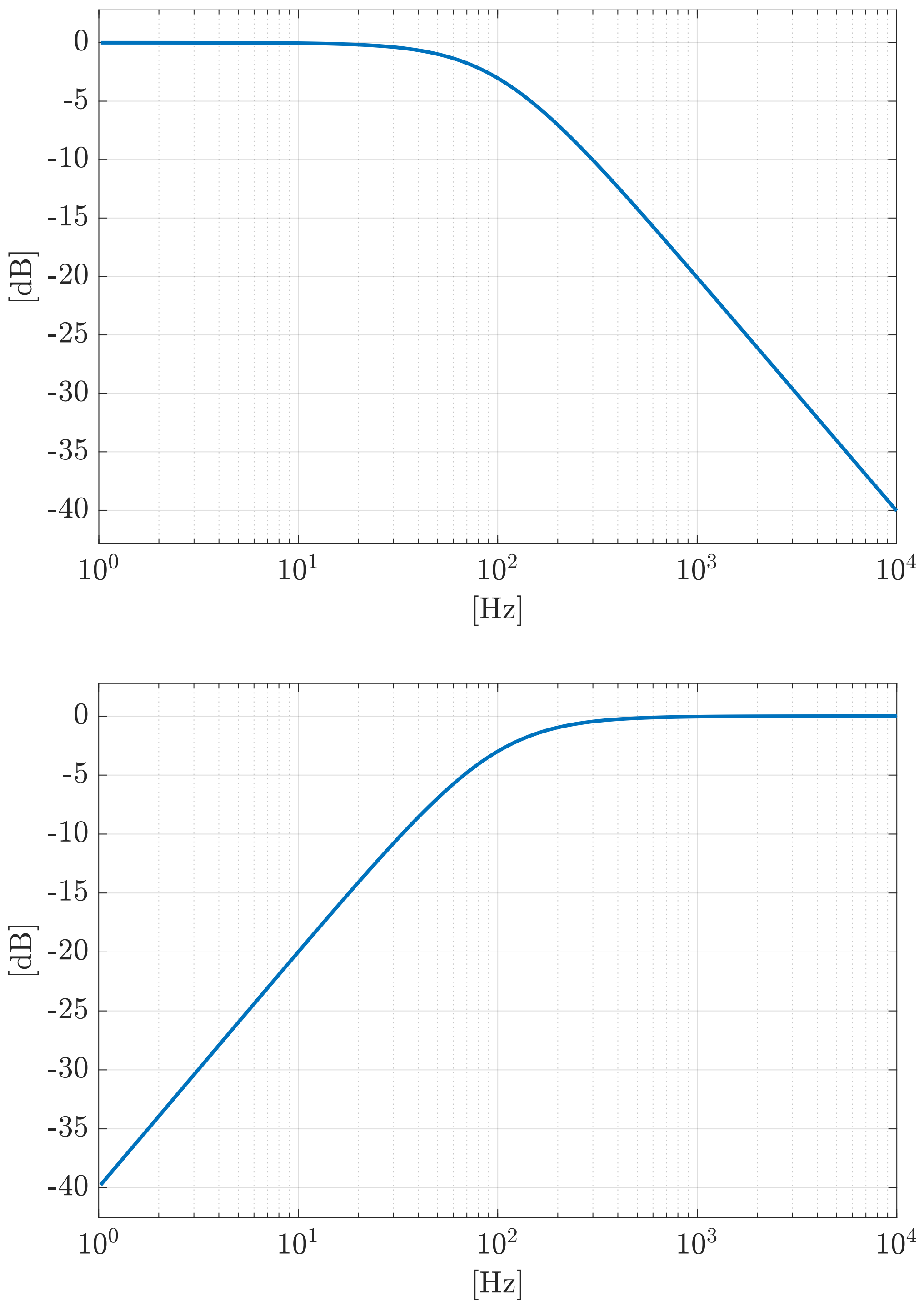


Figure 13.1: Low pass and high pass first order filters.

13.2 Second order filters

Second order filters are filters with as many as two zeros and two poles. Their transfer function has the form

$$T(s) = \frac{a_0 + sa_1 + s^2 a_2}{s^2 + s\frac{\omega_0}{Q} + \omega_0^2}$$

Where ω_0 is $\approx$ the angular frequency at which the poles are found, and Q, called *Quality factor*, is a measure of how close the poles are together. If we have complex conjugated poles $s = \alpha \pm j\omega$, their quality factor is $Q = \frac{1}{2}\frac{\omega}{\sqrt{\omega^2+\alpha^2}}$. Note that the poles (which are the values of s for which the denominator of $T(s)$ is zero) don't have to be complex conjugated. A filter with two real poles is still a second order filter.

Second order filters can be sized to obtain a large amount of types of filters. Sizing the filter means choosing all the parameters in the transfer function $Q, \omega_0, a_0, \ldots$ as to obtain the desired response.

Second order low pass filter

These are filters with two poles and no zeros, hence $T(s) = \frac{a_0}{s^2+s\frac{\omega_0}{Q}+\omega_0^2}$. They show a typical *peaking* in their transfer function around the frequency of the poles, and then decays with -40dB/dec, obtaining a much better cut-off effect than first order filters. The size of their peaking, in dB, is $\approx Q$. Figure 13.2 shows an example of this, which has been sized for 0dB gain and a 1kHz cut-off, meaning that ω_0 was chosen to be $\omega_0 \approx 2\pi 1$ kHz.

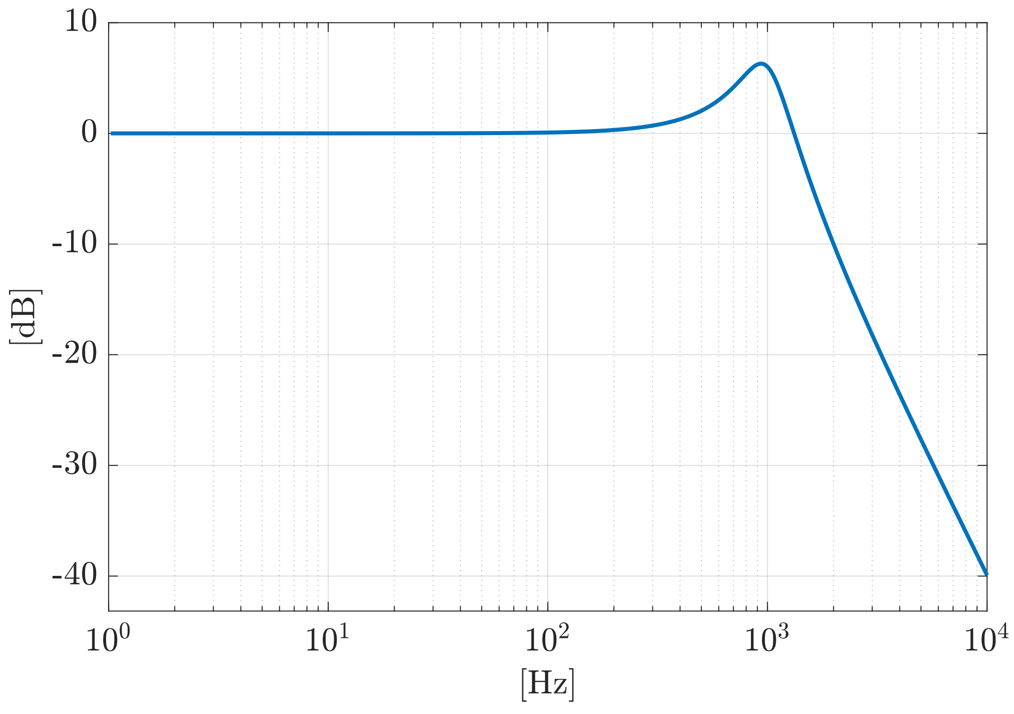


Figure 13.2: Second order low pass filter.

Second order high pass filter

A second order high pass filter has a +40dB/dec initial slope, which requires two zeros in the origin, and then a pair of poles to obtain a flat region at high frequency. The transfer function is of the kind $T(s) = \frac{s^2 a_2}{s^2+s\frac{\omega_0}{Q}+\omega_0^2}$. Figure 13.3 shows an example Bode plot for a filter with a 1kHz pole frequency.

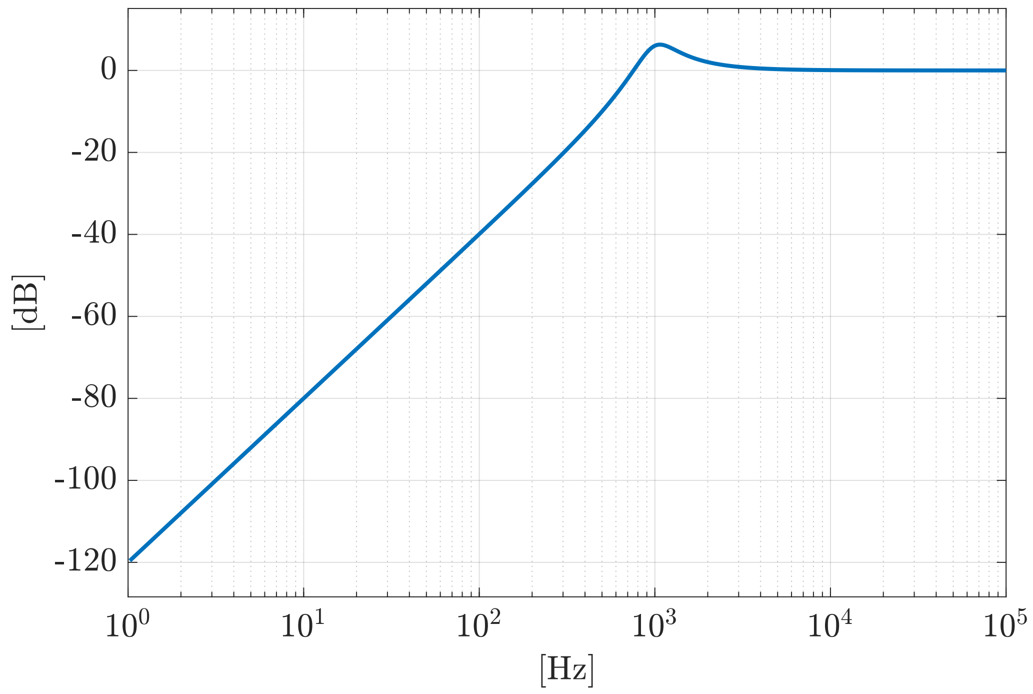


Figure 13.3: Second order high pass filter.

Band pass filter

A band pass filter is a type of filter that removes all frequency components outside a specific band of interest. It features a zero in the origin, a pole at the beginning of the pass region, and another pole at the end of the pass region. Their transfer function is thus of the kind $T(s) = \frac{s a_1}{s^2+s\frac{\omega_0}{Q}+\omega_0^2}$. As the Q factor is a measure of how close together the poles are, the higher the Q factor, the narrower the pass region will be. Figure 13.4 shows an example of this filter, sized with complex conjugated poles at around 1 kHz. Note the ±20dB/dec slopes at either sides.

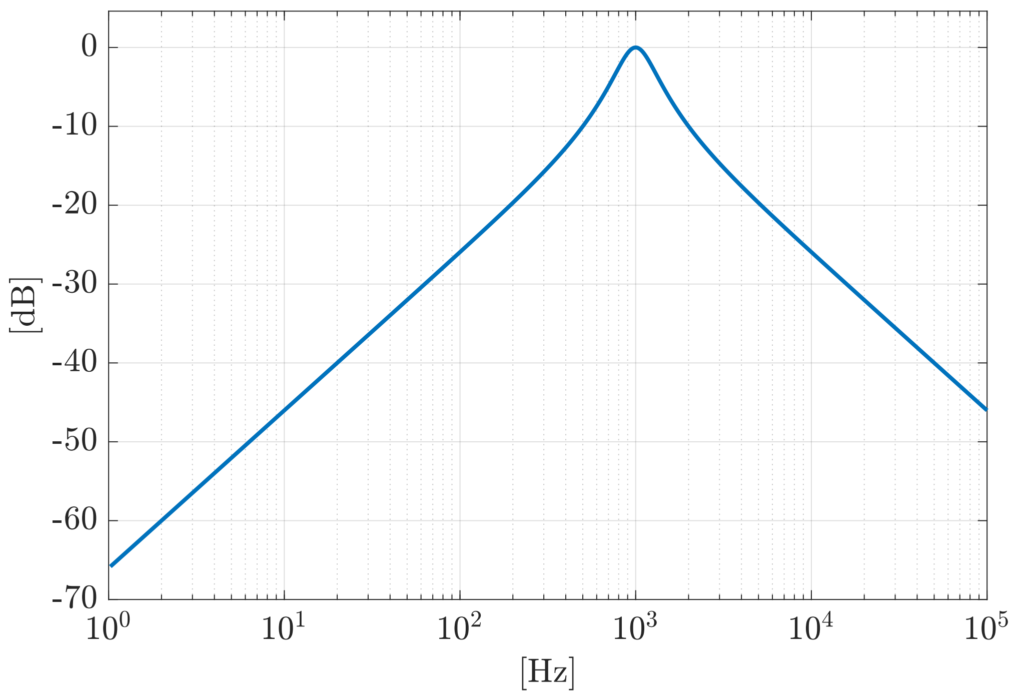


Figure 13.4: Second order band pass filter.

Notch filter

The notch filter is used to remove frequencies components at a pre-determined frequency. It can be useful, for instance, to remove disturbances whose frequency is known in advance. To achieve this, it has complex conjugated zeros with no real part ($z = \pm\omega_0 j$) at the frequency of interest, and complex conjugated poles around the same frequency. As the system draws closer to the target frequency, the poles activate alongside the zeros. As the zeros are on the imaginary axis, they show as actual 0 ($-\infty$ dB) on the Bode plot, which is, in fact, made for $s = 2\pi jf$ (on the imaginary axis). The activation of both zeros and poles at the same frequency allows to quickly recover the gain. The transfer function is thus $T(s) = \alpha\frac{\omega_0+s^2}{s^2+s\frac{\omega_0}{Q}+\omega_0^2}$. Figure 13.5 shows an example of this filter, designed to remove frequency components at around 1 kHz. Note that the zero should go to minus infinity dB, but, due to the limitations of the plotting code, this isn't the case.

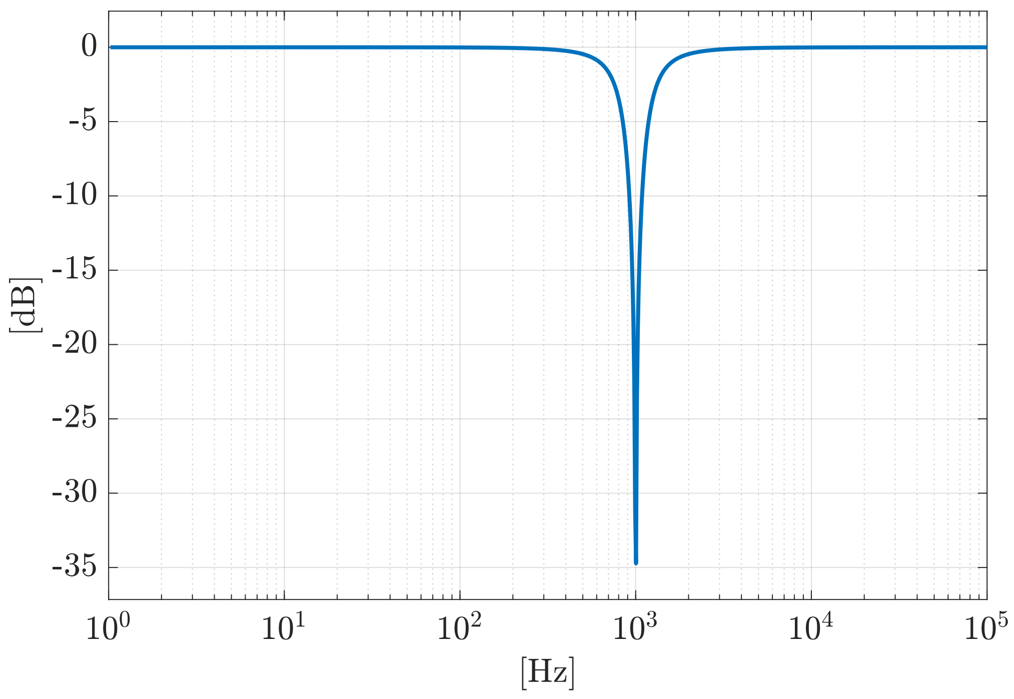


Figure 13.5: Notch filter.

Low pass notch filter

The low pass notch filter uses zeros on the imaginary axis, just like the Notch filter, to reject certain frequency components. However, the poles are placed at a frequency slightly below that of the zeros, thus allowing for their activation (and the consequent reduction of the gain) before the peak at $-\infty$ dB due to the imaginary zeros. The transfer function will be of the kind $T(s) = \alpha \frac{\omega_1 + s^2}{s^2 + s\frac{\omega_0}{Q} + \omega_0^2}$, with $\omega_0 < \omega_1$. Figure 13.6 shows an example of this filter, designed to reject a disturbances at 3 kHz while allowing signals below 1 kHz to pass unopposed. This means that the zeros have been put at 3 kHz and the poles at 1 kHz. Note that the low frequency and high frequency gains are not the same, as can be derived by taking the limits for $s \longrightarrow \infty$ and $s \longrightarrow 0$ of the transfer function.

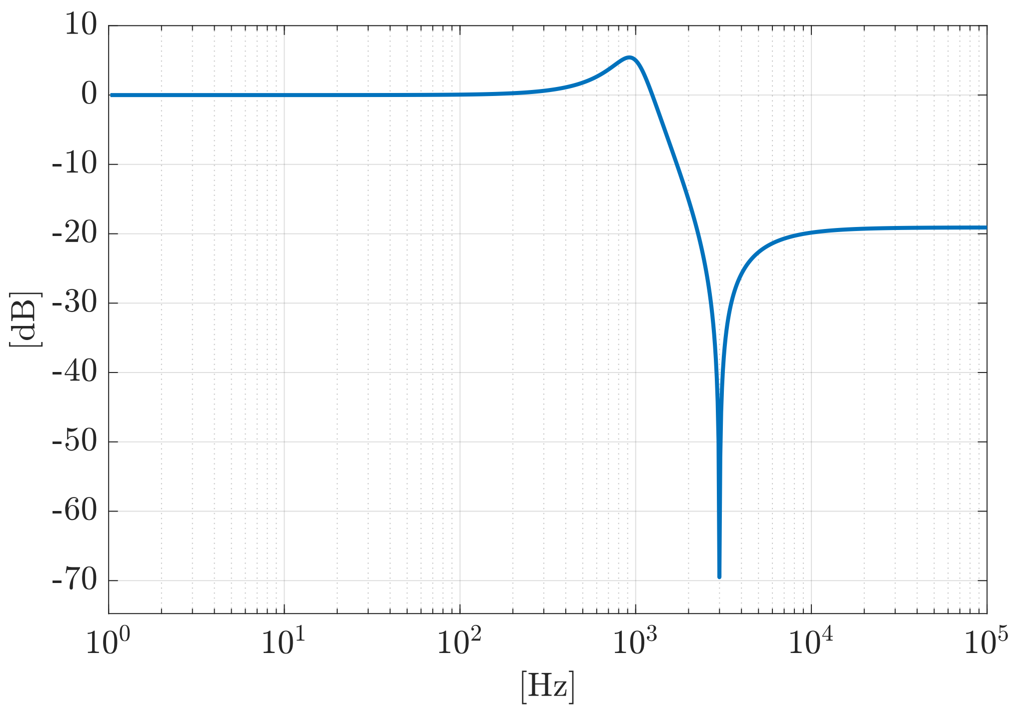


Figure 13.6: Low-pass notch filter.

High pass notch filter

The high pass notch filter is very similar to the low pass notch filter, but the poles are placed slightly above the frequency of the zeros. This means that we first reject the disturbance, then increase the gain due to the zeros, and last the poles activate. The transfer function is $T(s) = \alpha \frac{\omega_1 + s^2}{s^2 + s\frac{\omega_0}{Q} + \omega_0^2}$, with $\omega_0 > \omega_1$. An example is shown in figure 13.7, showing rejection at 400 Hz and a pass band starting at 1 kHz.

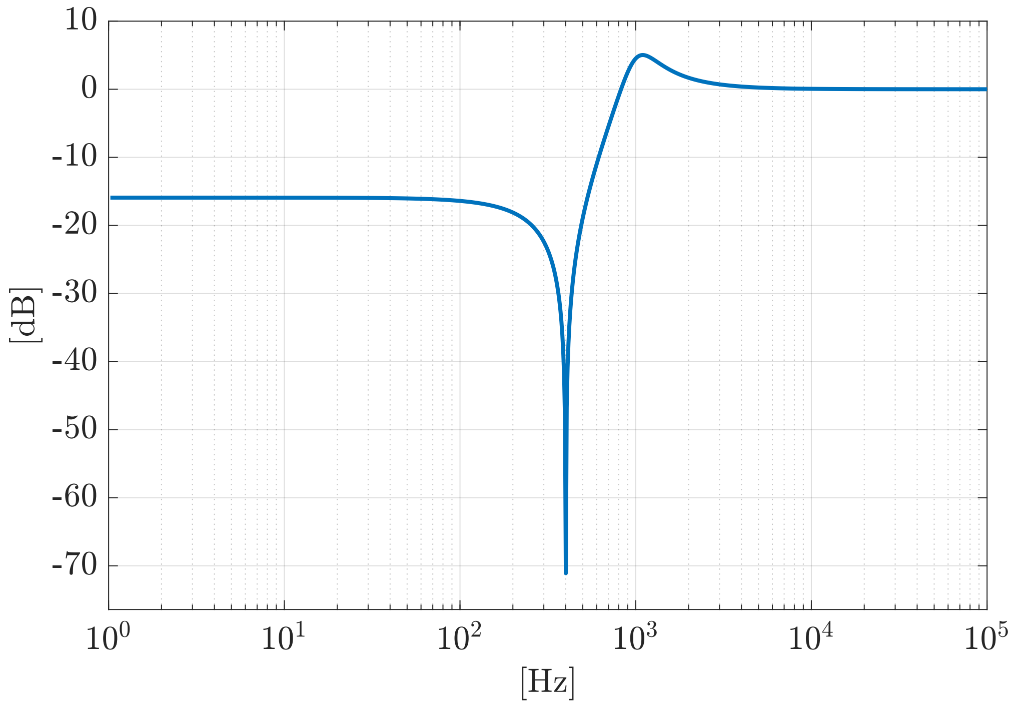


Figure 13.7: High-pass notch filter.

13.3 Implementations of second order filters

Implementing the filters seen above involves the creation of circuits with complex conjugated poles and zeros. This can be achieved in two ways: with feedback circuits, and with RLC circuits.

13.3.1 Sallen-Key cells

The idea of the Sallen-Key cells is to use an open loop transfer function with the appropriate poles and zeros so that the closed-loop transfer, whose poles are found by solving $G_{loop}(s) = 1$, is exactly the filter's transfer. Note that configurations with interacting zeros also allow for the zeros to be on the imaginary axis. The closed loop transfer can also be obtained by simply solving the circuit's equations, which allows us to obtain the complete transfer function and size the components. Note that depending on the filter to be obtained, which might show different zeros, different topologies are needed.

Low Pass Filter Sallen-Key

This is the most basic Sallen-Key cell. It shows two poles and no zeros in the closed loop transfer, and it's thus perfect for implementing the second order low pass filter. Figure 13.8

shows the design. The closed loop transfer function, instead, is

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{\frac{G}{R_1C1R_2C_2}}{s^2 + s(\frac{1}{R_1C_1} + \frac{1}{R_2C_1} + \frac{1-G}{R_2C_2}) + \frac{1}{R_1C_1R_2C_2}}$$

where $G = 1 + \frac{R_b}{R_a}$ is the gain of the non-inverting configuration within the loop of the stage. By inspecting this transfer function, any second order low pass filter can be constructed.

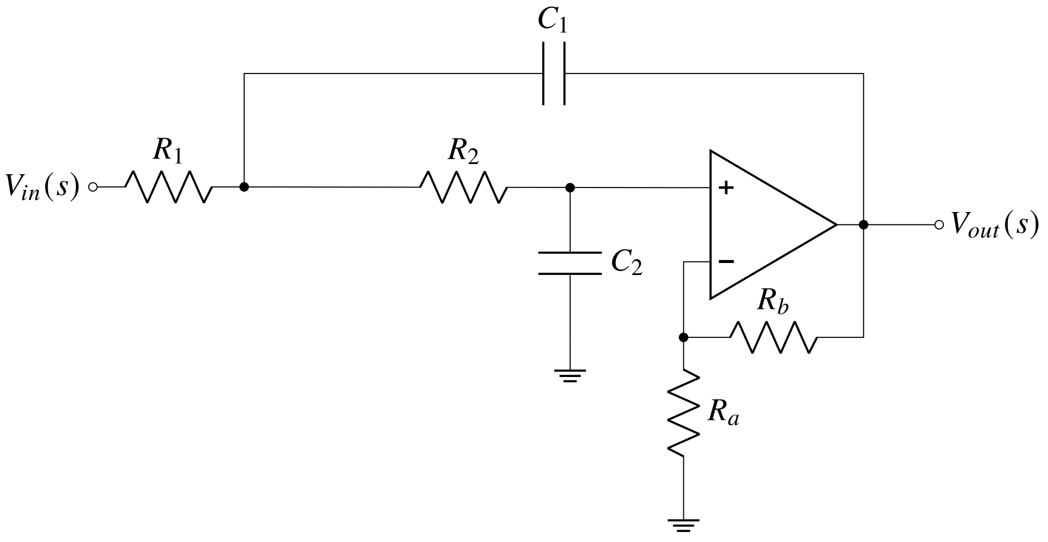


Figure 13.8: Sallen-Key Low Pass Filter Circuit.

High Pass Filter Sallen-Key

This cell was made to implement high pass filters. As such, it features two zeros in the origin (which require two capacitors between the input and output path), and two poles, given by the closed loop system. The design is shown in figure 13.9, which highlights how similar it is to the low pass cell: all that is needed is to swap the capacitors and resistors. The closed loop transfer function is

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{Gs^2}{s^2 + s(\frac{1}{R_2C_2} + \frac{1}{R_2C_1} + \frac{1-G}{R_1C_1}) + \frac{1}{R_1C_1R_2C_2}}$$

where $G = 1 + \frac{R_b}{R_a}$ is the gain of the non-inverting configuration within the loop of the stage. By inspecting this transfer function, any second order band high filter can be constructed.

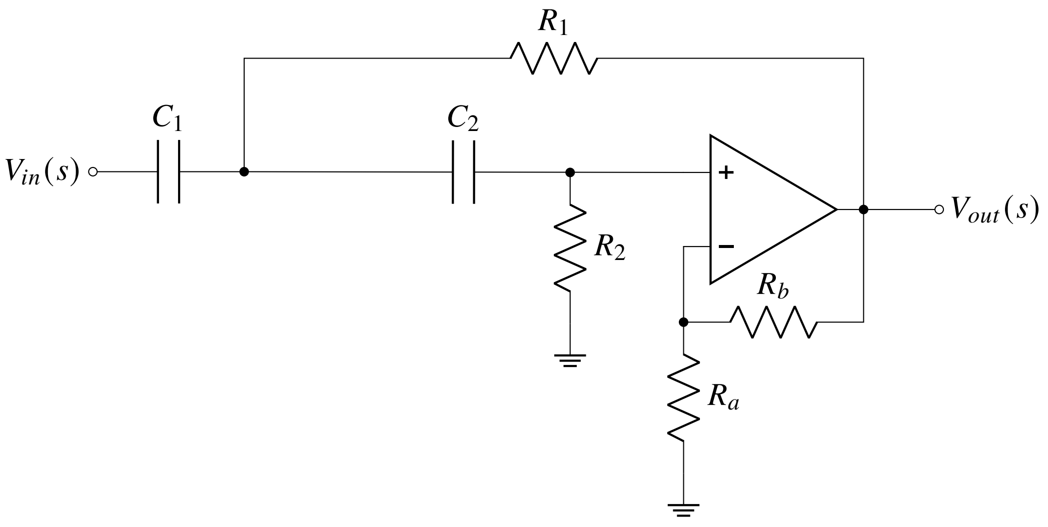


Figure 13.9: Sallen-Key High Pass Filter Circuit.

Band Pass Filter Sallen-Key

This cell (shown in 13.10) is intended to implement the band pass filter seen above. As such, it requires a zero in the origin (One capacitor in the input-output path), and two poles. The capacitor to ground ensures that high frequencies are eliminated. The closed loop transfer function is

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{s(\frac{G}{R_1C_1})}{s^2 + s(\frac{1}{R_1C_1} + \frac{1}{R_2C_1} + \frac{R_2}{C_2} - \frac{R_b}{R_aR_fC_1}) + \frac{R_1+R_f}{R_1R_fR_2C_1C_2}}$$

where $G = 1 + \frac{R_b}{R_a}$ is the gain of the non-inverting configuration within the loop of the stage. By inspecting this transfer function, any second order band pass filter can be constructed.

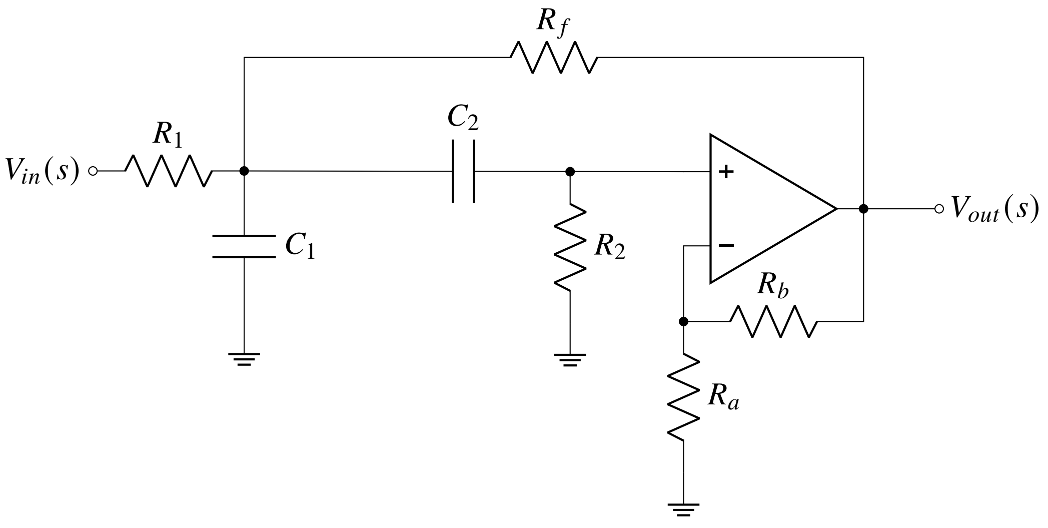


Figure 13.10: Sallen-Key Band Pass Filter Circuit.

Notch Filter Sallen-Key

The notch filter sallen key cell implements the notch filter seen above. As such, it requires two complex zeros and two complex poles in its closed loop transfer function. Obtaining complex zeros requires interacting capacitors in a very peculiar configuration. The design is shown in figure 13.11 and its closed loop transfer function is

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{G(s^2 + \frac{1}{R^2C^2})}{s^2 + 4\frac{1-G}{RC}s + \frac{1}{R^2C^2}}$$

where $G = 1 + \frac{R_b}{R_a}$ is the gain of the non-inverting configuration within the loop of the stage. By inspecting this transfer function, any second order notch filter can be constructed.

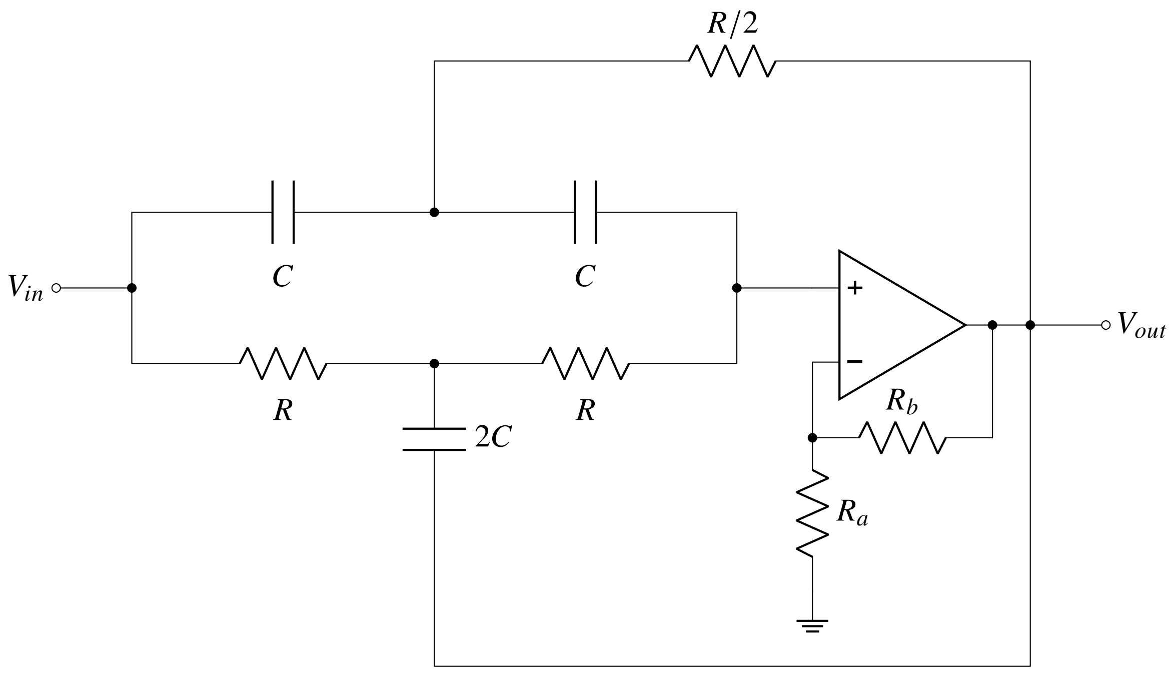


Figure 13.11: Sallen-Key Notch Filter.

13.3.2 Resonant RLC

Circuits featuring resistors, capacitors and inductors can show second order behaviors, with complex poles and zeros. Many such examples are shown in the literature to implement all different types of filters. As an example, the following circuit can implement a notch filter:

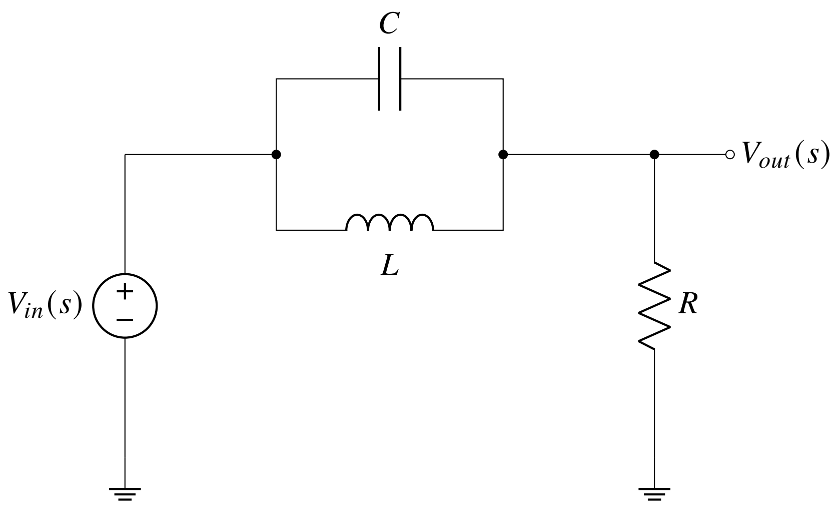


Figure 13.12: LRC Notch filter.

The transfer function can be readily obtained as a complex voltage partition,

$$V_{out} = V_{in}\frac{R}{R + Z_C(s)//Z_L(s)} = V_{in}\frac{R}{R + \frac{1}{sC}//sL} = \ldots = V_{in}\frac{R(1 + CLs^2)}{Ls + R(1 + s^2CL)}$$

Note the two complex zeros and the two poles, which is what's needed for a notch-type filter. Most other designs in the literature are also voltage partitions between complex impedances, and their transfers can be obtained in a similar manner.

13.3.3 Antoniou's gyrator

If we want to to use a LRC topology, but we cannot use a real inductor, for instance for an integrated circuit, there is a topology, called *Antoniou's gyrator* that can show an inductive impedance (like $Z_L(S) = L \cdot s$), while only employing capacitive impedances ($Z_C(s) = 1/(sC)$), resistors and opamps. The topology is shown in figure 13.13, and the input impedance seen is

$$Z_{in} = \frac{Z_1 \cdot Z_3 \cdot Z_5}{Z_2 \cdot Z_4} \tag{13.1}$$

We can see that if either Z_4 or Z_1 is a capacitor, the impedance seen will be inductive. To compute this formula, one needs to apply a test current, which will also flow into the opamps, and compute the voltage at the input node, taking their ratio. Note that due to the opamp's shorts only three unique potentials exists in the stage: V_L, V_A and V_B. By imposing that the currents into two adjacent impedances connected to the inputs of the opamps are the same, one can find the equations

$$\frac{V_C - V_L}{Z_4} = \frac{V_L}{Z_5}$$
$$\frac{V_L - V_A}{Z_1} = i_t$$
$$\frac{V_A - V_L}{Z_2} = \frac{V_L - V_C}{Z_3}$$

From which $V_L(i_t)$ can be derived, and from that we get the input impedance of equation 13.1.

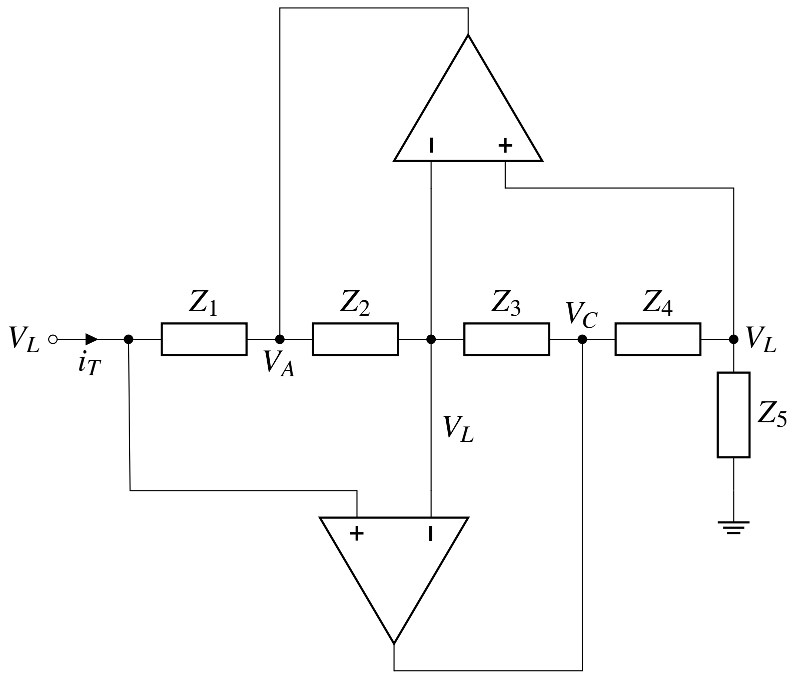


Figure 13.13: Antoniou's gyrator.

13.4 Higher order filters

There are several families of analog filters, each optimized for a different kind of processing, and each extendable to filters of order greater than two. Some have been optimized with the idea of removing certain frequency components more or less effectively, like we've seen for the second order filters above, and others to delay analog signals. Let's look at them one by one. Implementing higher order filters involves cascading several stages like the ones shown above for second order filters. Also, note that the filters are presented here in their low pass variation, but the high-pass versions exist and can be found by simply computing $1 - T(s)$, where $T(s)$ is the normalized (gain 1) transfer function.

13.4.1 Butterworth filters

The butterworth filters are a family of filters optimized for pass-band flatness. This means that all frequency components below the cut-off frequency are transferred with almost the same gain, independently of their exact frequency. The tradeoff needed to achieve this is a modestly sharp cutoff, which is only of -20n dB/dec, where n is the order of the filter. This makes butterworth filters widely used in audio and other high fidelity application, where minimal distortion is more important that achieving the sharpest possible cut-off.
The transfer function, for a filter with gain 1, is of the kind $T(s) = \frac{1}{D(s)} = \frac{1}{(s-s_1)(s-s_2)...(s-s_n)}$, and the poles are located on a semi-circle (remember that the frequency of a pole is $\frac{|s|}{2\pi}$), and are in general: $s_k = (2\pi f_c) \cdot e^{j\frac{(2k+n-1)\pi}{2n}} = \sigma + j\omega$, where n is the order of the filter, f_c is the desired cut-off frequency and $k = 1, 2, \ldots, n$. An example of where the poles should be for a 4th order filter is shown below in figure 13.14. The bode plot for the same filter is shown in figure 13.15.

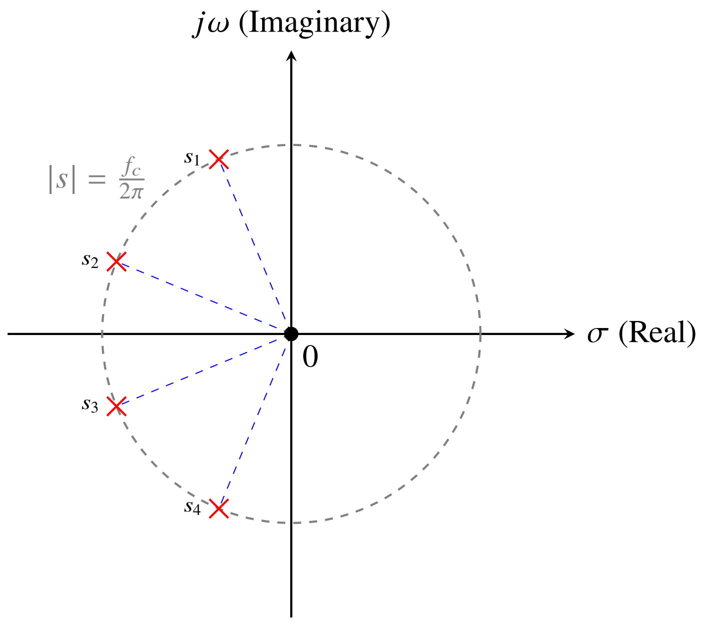


Figure 13.14: Poles of the Butterworth filter. Note that $\sigma < 0$ for stability.

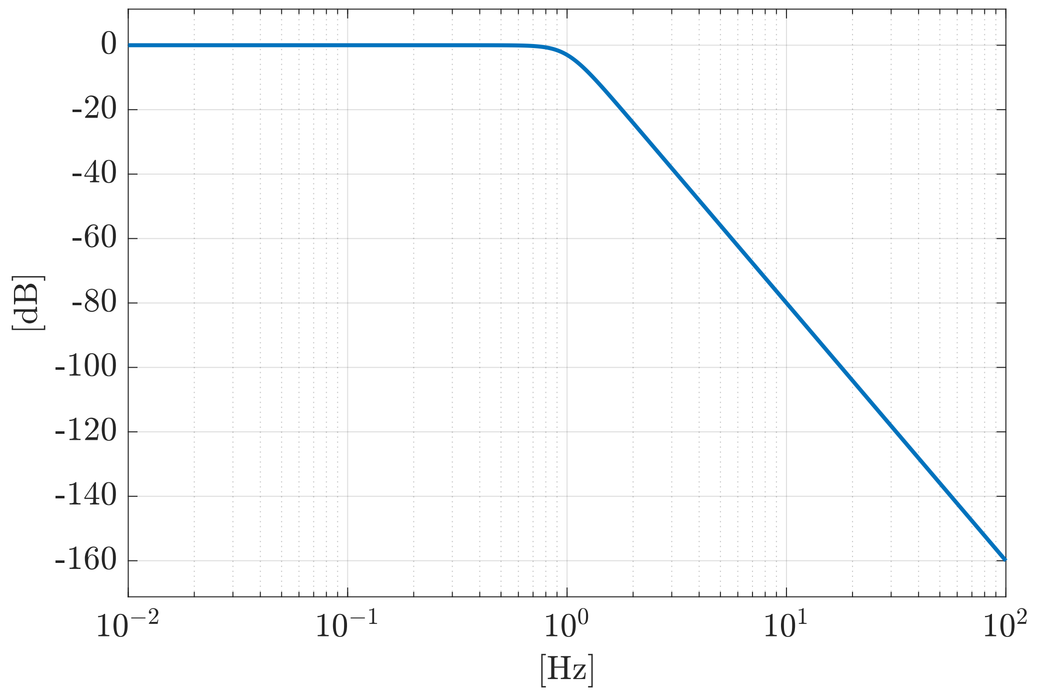


Figure 13.15: Bode plot of a 4th order butterworth filter.

13.4.2 Chebyshev filters

Chebyshev filters (pronunciation like cebicef) offer a much sharper cut-off, allowing them to eliminate frequency components much closer to the cut-off frequency much more effectively than butterworth filters. This means that to achieve a given reduction in a disturbance close to the cut-off a simpler, cheaper circuit is needed, as the order of the filter directly influences the complexity of the circuit implementing it. The tradeoff needed to achieve this is the presence of ripples in the frequency response (see figure 13.17). This means that signals with a broad band below the cut-off frequency are distorted, as their components at different frequencies are amplified differently. The transfer function's structure is like that of Butterworth filters, $T(s) = \frac{1}{D(s)} = \frac{1}{(s-s_1)(s-s_2)...(s-s_n)}$, but the poles are located on an *ellipse*. This means that their frequency on the bode plot, which is $\frac{|s|}{2\pi}$, won't be the same for all. This is ultimately the reason for the presence of so many ripples in the Bode plot. The number of ripples is equal to the order of the filter.

Order (n)	**Denominator Polynomial** $D(s)$
1	$s + 1.9652$
2	$s^2 + 1.0977s + 1.1025$
3	$(s + 0.4942)(s^2 + 0.4942s + 0.9942)$
4	$(s^2 + 0.2791s + 0.9865)(s^2 + 0.6737s + 0.2794)$
5	$(s + 0.2895)(s^2 + 0.1789s + 0.9883)(s^2 + 0.4684s + 0.4293)$
6	$(s^2 + 0.1244s + 0.9907)(s^2 + 0.3398s + 0.5577)(s^2 + 0.4642s + 0.1247)$
7	$(s + 0.2054)(s^2 + 0.0919s + 0.9927)(s^2 + 0.2566s + 0.6535)(s^2 + 0.3485s + 0.2304)$
8	$(s^2 + 0.0708s + 0.9941)(s^2 + 0.2003s + 0.7230)(s^2 + 0.2853s + 0.3330)(s^2 + 0.3161s + 0.0710)$

Table 13.1: Chebyshev filter polynomials (1 dB ripple). The numerator of the transfer function should be adjusted to obtain the desired gain. The position of the poles on the complex plane are shown in figure 13.16 for a 6th order filter.

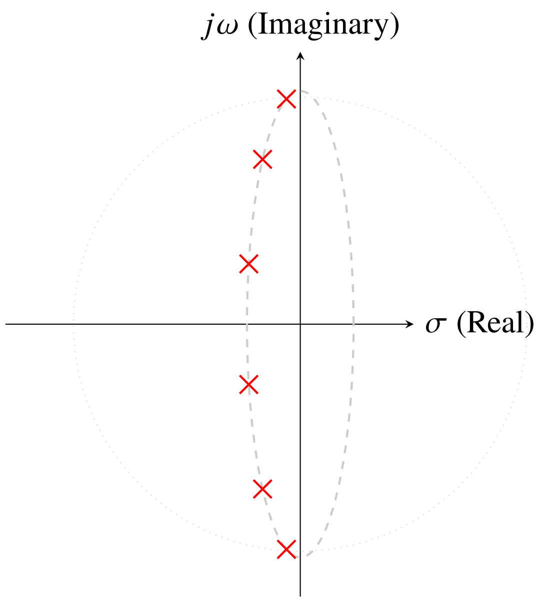


Figure 13.16: Poles of the 6th order Chebyshev filter.

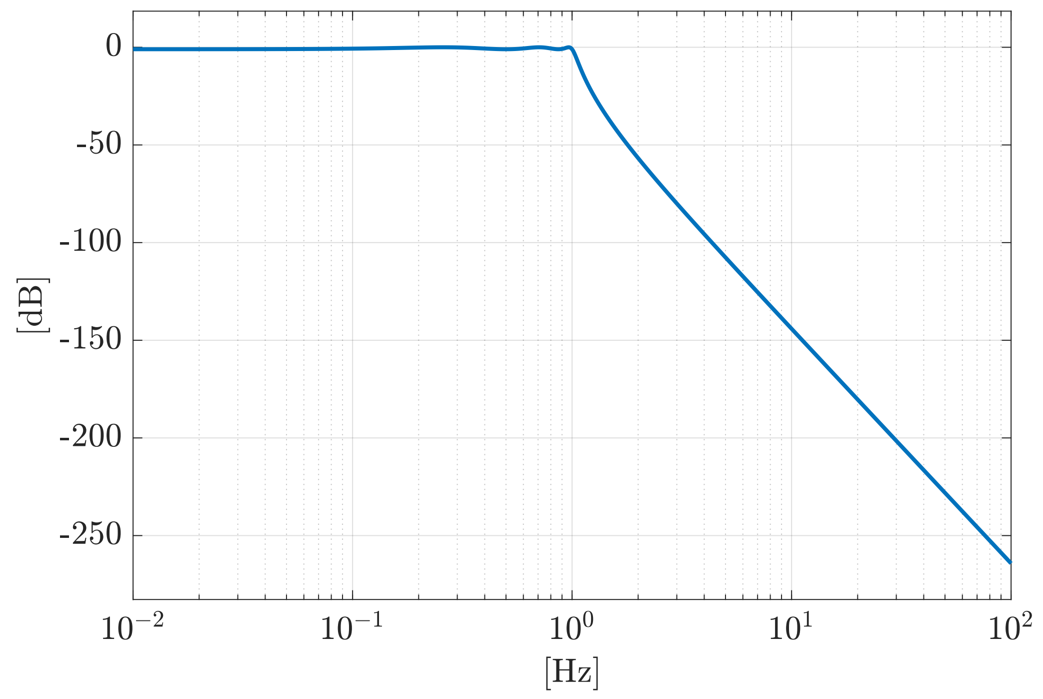


Figure 13.17: Bode plot of the 6th order Chebyshev filter. Notice ripples and diving behavior.

Note that the final slope of Chebyshev filters will still be -20n dB/dec, like for Butterworth filters. The more aggressive cut-off is only seen close to the cut-off frequency, where the Chebyshev response dives faster.

13.4.3 Inverse Chebyshev filters

The inverse Chebyshev filters have been introduced to address the primary shortcoming of the Chebyshev filter: the ripples in the pass-band distorting signals. To this aim, zeros have been introduced on the imaginary axis, at a frequency above the cut-off. These zeros ensure complete rejection of signals at their frequencies above the cut-off, and remove all ripples from the pass-band. This means, however, that more ripples appear in the cut-off band. Usually this isn't a problem, as the disturbances at those frequencies are eliminated anyways. The transfer function will show both poles and complex zeros, $T(s) = T_0 \frac{(s^2+\omega_1^2)(s^2+\omega_2^2)...(s^2+\omega_m^2)}{(s-s_1)(s-s_2)...(s-s_n)}$. Note that the positions of the zeros and poles depend on the desired stop-band gain (attenuation). Table 13.2 shows the locations of these singularities to achieve a -20dB gain in the stop band. Figure 13.18 shows the frequency response of a 4th order filter. Note that the number of $-\infty$ dB peaks is equal to the number of zeros divided by two, as the other twos are at negative frequencies.

Order (n)	**Poles** ($\sigma \pm j\omega$)	**Zeros** ($j\omega$)
1	-1.0000	None
2	$-0.7071 \pm j0.7071$	$\pm j1.4142$
3	-0.9562	$\pm j1.1547$
	$-0.4781 \pm j0.8281$	
4	$-0.4276 \pm j0.9324$	$\pm j1.0824$
	$-0.8543 \pm j0.3015$	$\pm j2.6131$
5	-0.9123	$\pm j1.0515$
	$-0.3761 \pm j0.9542$	$\pm j1.7013$
	$-0.7382 \pm j0.4356$	
6	$-0.3341 \pm j0.9754$	$\pm j1.0353$
	$-0.6234 \pm j0.6213$	$\pm j1.4142$
	$-0.8125 \pm j0.2114$	$\pm j3.8637$

Table 13.2: Inverse Chebyshev Poles and Zeros for −20 dB stopband attenuation.

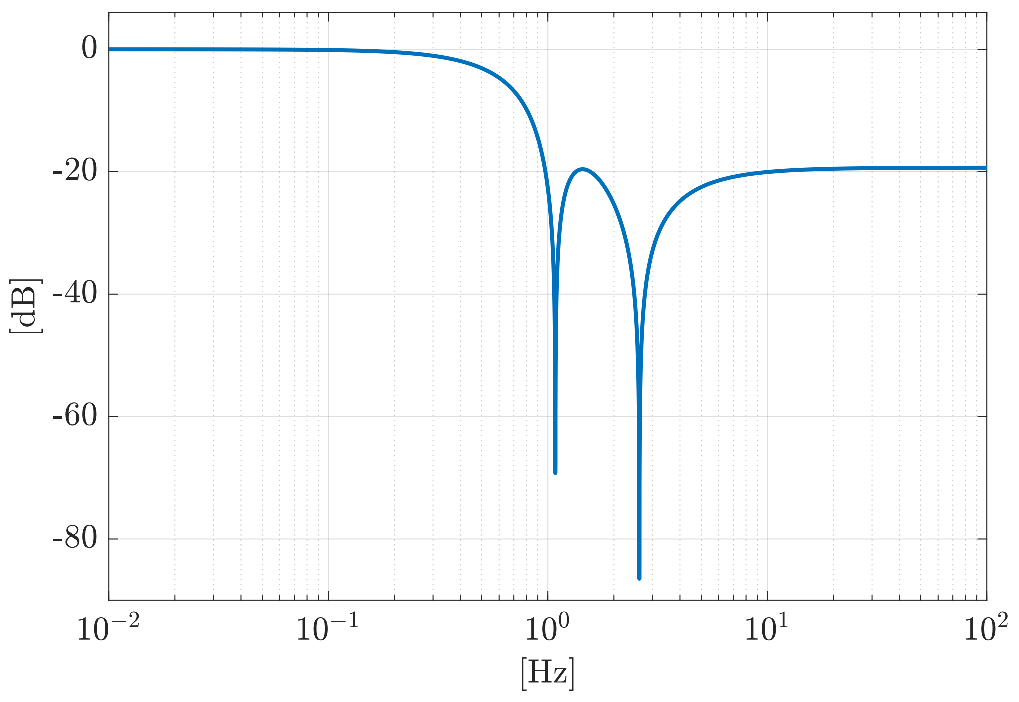


Figure 13.18: Bode plot of the 4th order inverse Chebyshev filter. Notice ripples and pass-band flatness.

Notice how flat the response is in the pass-band: these filters are suited for high quality applications requiring minimal distoritions, unlike the normal Chebyshev filters.

13.4.4 Bessel filters

Bessel filters are designed to delay a broadband signal without distorting it, like a square wave or anything else requiring delayed response. For this reason, they must have a linear phase-frequency relation. To understant this point, let's imagine an input signal $\sin(\omega t)$. The output of the filter will be $\sin(\omega t + \phi)$, where ϕ is the phase of the transfer function at a frequency ω. If we want to pretend that that ϕ looks like a delay in time, we could rewrite the output like this: $\sin(\omega(t + \frac{\phi}{\omega}))$. Now we can see that the signal has been delayed in time by ϕ/ω. As this delay depends on frequency, a signal made up by multiple harmonics at different frequencies would be distorted (as each component gets delayed by a different amount), unless the phase of the transfer function, ϕ, also changed linearly with frequency. If we imagine a relation of the kind $\phi(\omega) = -\gamma\omega$, the output simplifies to $\sin(\omega(t + \frac{\phi}{\omega})) = \sin(\omega(t - \gamma))$. Now the harmonic has been delayed in time by an amount γ depending on the steepness of the phase-frequency relation. This amount doesn't depend on ω, which means that, as long as the phase is linear with frequency, a broadband signal would see all of its harmonics delayed by the same amount of time. For this reason, it's resonable to define the *group delay* of the filter as $\tau = -\frac{d\phi(\omega)}{d\omega}$. It's possible to prove that to achieve the most linear phase-frequency relation possible the transfer function must be $T(s) = \frac{d_0}{d_0 + sd_1 + s^2 d_2 + \ldots + d_n s^n}$, with $d_k = \frac{(2n-k)!}{2^{n-k} k!(n-k)!}$, where n is the order of the filter and $k = 1, 2, \ldots, n$. The frequency response of a 4th order Bessel filter is shown below in figure 13.19. The figure highlights how Bessel filters only offer a linear phase response (and thus constant group delay) for a portion of the frequency spectrum. The filter must be sized based on the frequency of the signal we want to delay. Note that due to its proprety, the Bessel filter is the only filter seen here that doesn't show overshoots in its step response. This fidelity is what makes it pretty useful for digital signals.

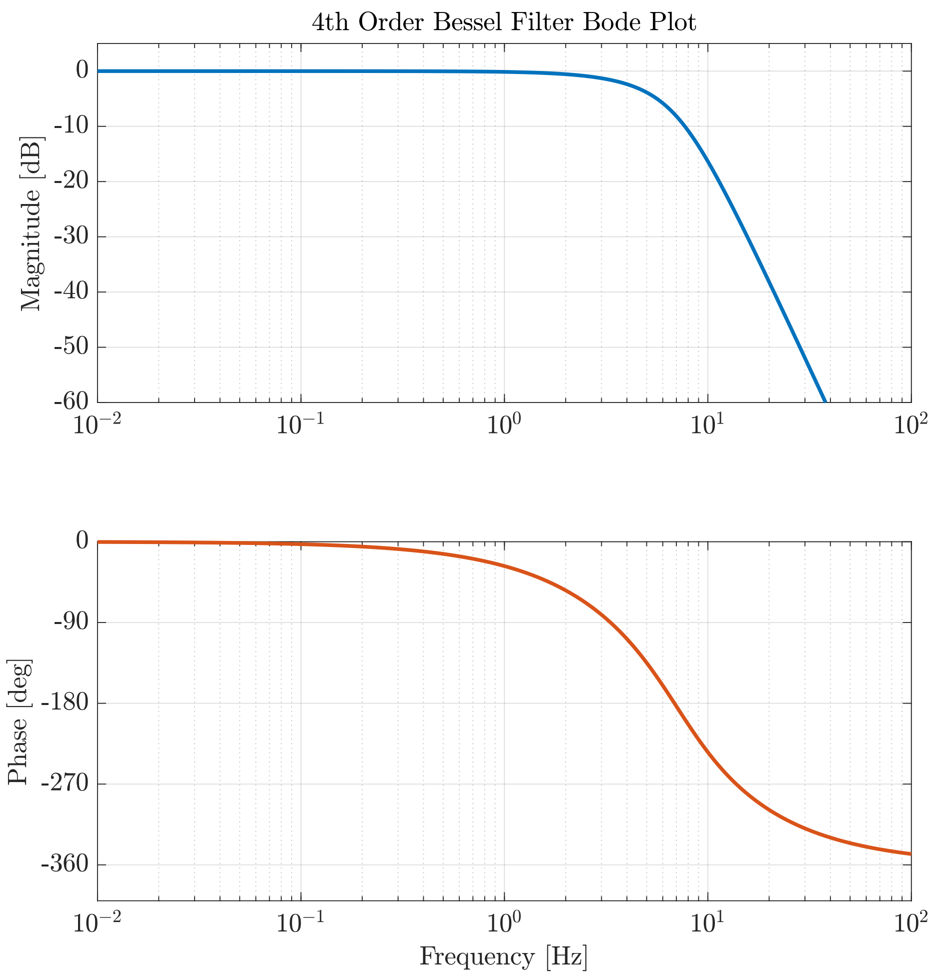


Figure 13.19: Bode plots of a 4th order Bessel filter. Notice linearity of the phase in the roll-off zone.

13.4.5 Elliptical filters

The elliptical filter is a filter showing the fastest possible cut-off, showing oscillations in its frequency response both before and after the cut-off frequency. In this regard it's a sort of combination between the two Chebyshev filters, and it's only really used for applications requiring the sharpest possible separation between the pass-band and the cut-off band. For instance, it may be useful as a anti-aliasing filter when sampling close to the Nyquist limit. The poles and zeros for an elliptical filter with normalized cut-off frequency, -40dB attenuation and 1dB ripples in the pass band are given below:

Order (n)	**Poles** ($\sigma \pm j\omega$)	**Zeros** ($j\omega$)
1	-0.3888	None
2	$-0.2465 \pm j1.0416$	$\pm j2.3451$
3	-0.2114	$\pm j1.4554$
	$-0.0964 \pm j1.0422$	
4	$-0.0482 \pm j1.0347$	$\pm j1.2014$
	$-0.1764 \pm j0.3951$	$\pm j3.6558$
5	-0.1415	$\pm j1.1072$
	$-0.0275 \pm j1.0264$	$\pm j2.0011$
	$-0.1182 \pm j0.6251$	
6	$-0.0175 \pm j1.0203$	$\pm j1.0652$
	$-0.0764 \pm j0.7321$	$\pm j1.4231$
	$-0.1245 \pm j0.2541$	$\pm j5.8541$

Table 13.3: Elliptic Filter Poles and Zeros (1 dB Passband Ripple, −40 dB Stopband Attenuation).

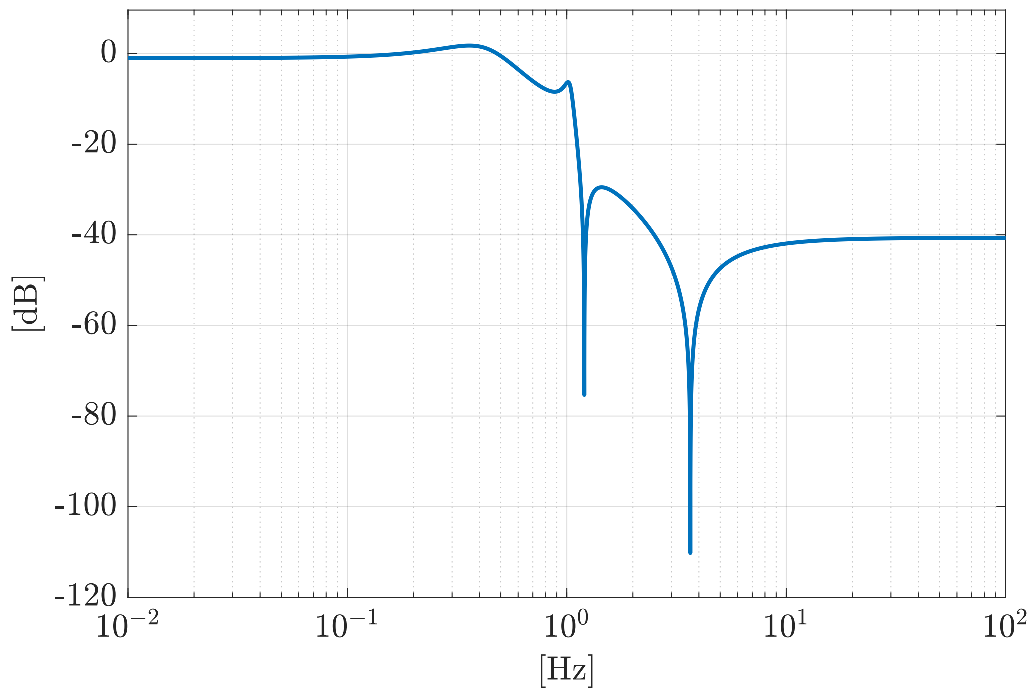


Figure 13.20: Bode plots of a 4th order elliptical filter. Notice the sharp transition and ripples in the pass band equal to the order of the filter.