	Linear Opamp 02 : Follow me
This document explains the follower circuit. That humble circuit yields a lot of information about opamp stability concepts.	
BOM	1x Dual Opamp MCP6002 Resistors: 10 k Ω , 33 k Ω , 100 k Ω and 330 k Ω Capacitors: 1 nF and 10 nF

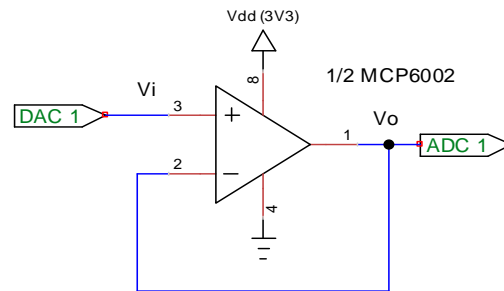
V1.0(24/3/2017) © Vicente Jiménez. License information is at the end of the document

Index

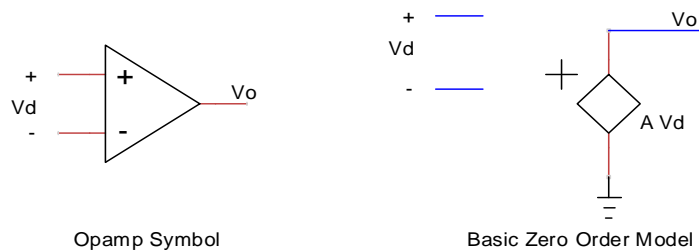
The Follower Circuit	2
Testing the follower	4
Follower on the roller coaster.....	7
The ideal first order opamp model	11
Frequency response of the follower circuit	14
The Gain-Bandwidth product	16
Going over the edge.....	17
Response of the two pole system	19
Last comments	21
References.....	22

The Follower Circuit

In the previous document we have seen the opamp operate in open loop configuration. Now we will see the follower circuit. In this circuit, an external input V_i , generated by DAC 1, is connected to the positive input (+) and the output of the opamp V_o , read by ADC 1, is feedback to the negative input (-).



We will try to deduce the behavior of the circuit using the zero order model.



We know that the zero order model is not good enough to model a real opamp because of saturation. However, we can rely on the model result as long as it gives a result between the saturation voltages. So, if using the zero order model we obtain a solution that satisfies:

$$V_{o \min} < V_o < V_{o \max}$$

Then we know that the more complex saturation model would have given the same result. That's a typical method to work in engineering:

- ❶ We take the easier model we can find. This model relates to a list of requirements to meet so that the results are valid.
- ❷ We use the model to solve a problem.
- ❸ We check that the solution meets all the requirements of the model.

Sometimes you are tempted to perform steps ❶ and ❷ and neglect to perform step ❸ . Using a model outside of its requirements can give us completely wrong results. So never neglect the third step.

In the case of the zero order model the model equations and the requirements are:

Zero Order Opamp Model	
Model Equations	Model Requirements
$V_d = V_{(+)} - V_{(-)}$ $V_o = A \cdot V_d$	$V_o > V_{o \min}$ $V_o < V_{o \max}$


We have a model and we have a circuit so we can solve the $V_o(V_i)$.

From the circuit:

$$V_{(+)} = V_i \quad V_{(-)} = V_o$$

Adding the model equations we get:

$$V_o = A(V_i - V_o)$$



1 Calculate $V_o(V_i)$ from the above equation. It can only depend on A.
Obtain the result for an ideal opamp that satisfies $A \rightarrow \infty$.

At this point you should know why this circuit is called the follower circuit.

The above calculation, although not difficult by any means, can be greatly simplified. Having an infinite gain means that, in order for the output to have a finite value, the input shall be zero.

$$V_d = \frac{V_o}{A} \Big|_{A \rightarrow \infty} = 0 \quad \text{if } V_o \text{ is any real number}$$

This is known as **virtual short circuit**. It is **short circuit** because nodes (+) and (-) are at the same voltage. It is **virtual** because it is not real, as no current can flow from one node to the other. The virtual short circuit is deduced from the ideal opamp model so it is valid when the model is valid. As we already know, the model is not always valid, like on saturated regions, so the virtual short circuit cannot always be used.

In fact, we know that the results won't be valid if the model predicts an output voltage out of the saturation limits.

If **virtual short circuit holds**, differential voltage is zero; it turns out that the output voltage V_o shall be the same as the input V_i . It cannot get easier than that. This result should be valid if we don't reach saturation.

Testing the follower

Now it is time to test the follower. Mount the circuit and perform a $V_o(V_i)$ curve using the **curveVV** command. Remember that the **curveVV** is in the DC module, so we need to import both the **slab** and the **slab_dc** modules.

```
>>> import slab
>>> import slab_dc as dc
>>> slab.connect()
>>> dc.curveVV(0,3.2,0.1)
```



2

Perform the $V_o(V_i)$ measurement of the follower circuit.
Does it work as expected?

A follower takes an input voltage and outputs an equal voltage. The range of voltages that a follower can follow depend on two conditions:

- The range of voltages it can provide on the output
- The range of voltages it can read on the input

We know from the previous document that the **Maximum Output Voltage Swing** parameter indicates the range of voltages that can be made available on the output.

Parameters	Sym	Min	Typ	Max	Units	Conditions
Output						
Maximum Output Voltage Swing	V_{OL}, V_{OH}	$V_{SS} + 25$	—	$V_{DD} - 25$	mV	$V_{DD} = 5.5V$, 0.5V Input Overdrive

In a similar way, the **Common Mode Input Range** parameter defines the input voltage range. It is named Common Mode because, when the virtual short circuit operates, both inputs have the same common voltage. In fact, the common voltage V_{CM} is defined:

$$V_{CM} = \frac{V_{(+)} + V_{(-)}}{2}$$

When the opamp operates on the linear region, outside of the saturation zones, the virtual shortcircuit usually holds so:

$$V_{CM} \approx V_{(+)} \approx V_{(-)}$$

The **Common Mode Input Range** parameter can be found on the MCP6002 datasheet:

Parameters	Sym	Min	Typ	Max	Units	Conditions
Common Mode Input Range	V_{CMR}	$V_{SS} - 0.3$	—	$V_{DD} + 0.3$	V	

We can see that the common mode input range not only includes the supply rails but also goes 0.3 V outside them.

As both the input and the output can limit the follower operation, being the output range smaller, it is the parameter that limits the follower range.

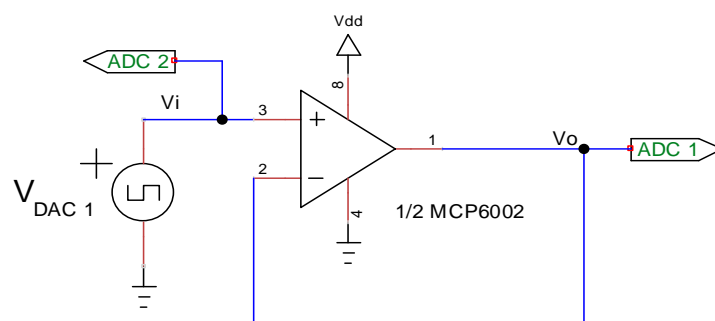
When, as happens on the MCP6002, both the Common Mode Range and the Output Swing go near both supplies we say that the opamp feature **Full-Rail Input and Output** operation. This is a very convenient feature for opamps that are powered at low voltages, as 3V, because the supply range is so small that we cannot lose any fraction of it.

In general you should not try to set the input voltages of an opamp outside of the supply range. The **Absolute Maximum Ratings** warn about that:

Absolute Maximum Ratings †	
$V_{DD} - V_{SS}$	7.0V
Current at Analog Input Pins (V_{IN+} , V_{IN-})	±2 mA
Analog Inputs (V_{IN+} , V_{IN-}) ††	$V_{SS} - 1.0V$ to $V_{DD} + 1.0V$
All Other Inputs and Outputs	$V_{SS} - 0.3V$ to $V_{DD} + 0.3V$

We can see that the opamp inputs $V_{(+)}$ and $V_{(-)}$ shall not go more than 1V outside of the supply range or the device could be permanently damaged.

The follower does not only work on DC, we can also test it in AC operation. We will test the circuit providing an input square wave with DAC 1 as shown in the circuit below. We will keep ADC 1 at V_o but we will add the ADC 2 input to V_i .



We want to test the opamp with a square wave. We will generate this wave with values between 1V and 2V using 100 data points.

```
>>> slab.waveSquare(1,2,100)
```

The module will respond something like:

```
100 point wave loaded
Wave frequency must be between 0.000100 and 400.00 Hz
Current frequency is 10.0 Hz
Remaining buffer space is 19900 samples
```

The module informs the frequency limits for the wave on the connected Hardware Board. It also gives current wave frequency for the current sample frequency that is 1 kHz by default. The hardware board uses an unified memory for samples. The board had a memory of 20000 samples. As 100 samples are used for the wave, we still have space for 19900 samples.

Now we set the frequency to 100 Hz.

```
>>> slab.setWaveFrequency(100)
```

That recalculates the sample information from the transient measurements. It returns current sample time T_s . As we want 100 points per cycle in a 100 Hz wave, the sample frequency f_s and the sample time T_s can be calculated:

$$f_s = 100 \cdot 100 \text{ Hz} = 10 \text{ kHz} \quad T_s = \frac{1}{f_s} = 100 \mu\text{s}$$

When we perform the measurement the system will generate the wave sending samples, one each 100 μs to draw the waveform.

We need to instruct the SLab module the number of samples to record. In order to record 5 full waves we need 500 points. We also need to indicate the number of ADCs to read, 2 in our case.

```
>>> slab.setTransientStorage(500,2)
```

As this is a long command, you can use a shorter **alias**:

```
>>> slab.tranStore(500,2)
```

Now we can perform the measurement and show the results.

```
>>> slab.wavePlot()
```

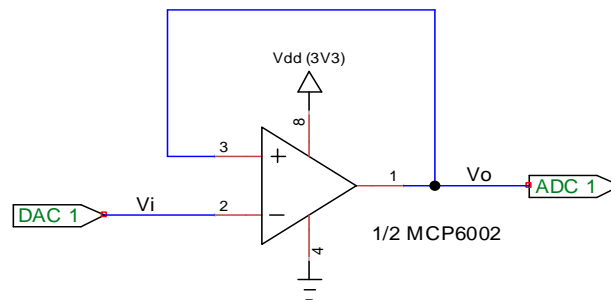


3

Perform the transient measurement.
Do the output follow the input?

Follower on the roller coaster

Considering all the explanations about the opamp we have made so far, connecting the potentiometer to the (-) input and the output to the (+) input should also work.



4

Obtain $V_o(V_i)$ using the Zero Order Opamp Model.

Obtain the result for an ideal opamp that satisfies $A \rightarrow \infty$.



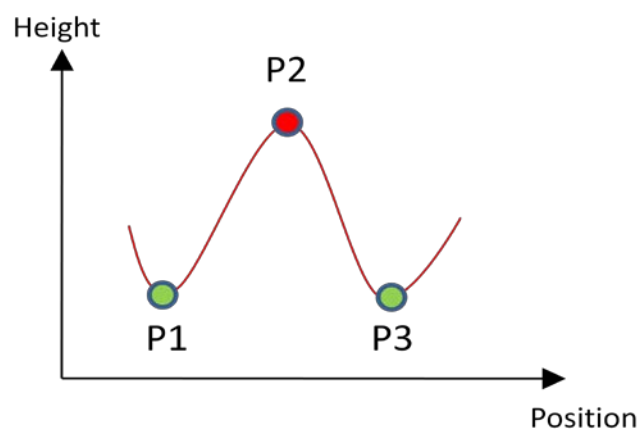
5

Mount the circuit as shown in the figure as shown in the figure and obtain the DC $V_o(V_i)$ curve.

Do the circuit work like before? Which are the changes?

As you can see the circuit doesn't work as before. The reason why it doesn't work is a question of **stability** and is related to the time response of the amplifier.

A typical example of stability is a rollercoaster. Imagine a [roller coaster](#) with peaks and valleys. In the following figure we see one peak at point P2 and two valleys at points P1 and P3.



P1 and P3 are two **stable** points. If you put a train in a valley and you push it to one side, the gravity will make it return to the valley. Once a train stops in the valley, you need to provide enough energy to reach a near peak to take it out of the valley.

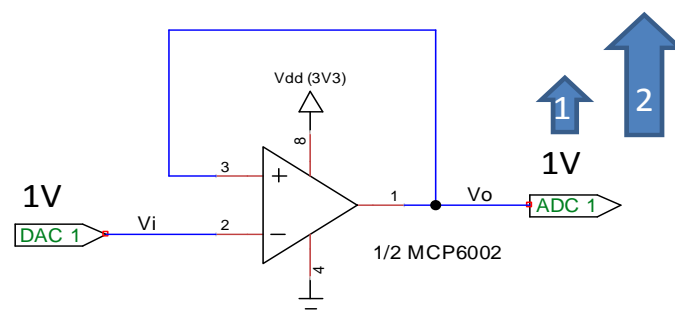
We say that in points P1 and P3 the system provides **negative feedback** because it tries to move the train in direction that is opposite to the disturbance force we are applying.

P2 is an **instable** point. Just at the top of a peak, the train can be still, but if you give a gentle push to any side, the gravity will do the rest to move the train far from this point.

In this case we say that the system provides **positive feedback** because it enhances any action we make on the train at this point.

Our bad follower circuit doesn't work because we are applying **positive feedback** and it is easy to see why. Suppose we have circuit V_i , connected to (-), at a voltage of 1V. The equilibrium condition in the linear region, as the virtual short circuit shows, will put the output node V_o at 1V also.

$$V_{iEQ} = 1V \quad V_{oEQ} = 1V + V_{dEQ} \approx 1V$$



Now, we perturbate the equilibrium by increasing V_o , as this voltage is applied to the positive input $V_{(+)}$, the differential voltage increases:

$$V_{(+)} = 1V + \Delta V \quad V_d = V_{dEQ} + \Delta V$$

That yields a voltage change in the output:

$$V_o = A(V_d) = A(V_{dEQ} + \Delta V) = A \cdot V_{dEQ} + A \cdot \Delta V = 1V + A \cdot \Delta V$$

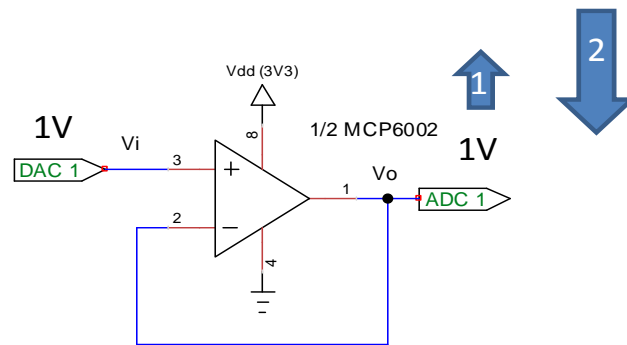
So, output voltage will increase more. We had a ΔV perturbation and now we have a much higher $A \cdot \Delta V$ perturbation. That perturbation will race towards greater voltages and will only end when V_o reaches the maximum output voltage it can provide. This end point, at the V_{Omax} saturation voltage, is quite stable as V_o cannot go higher.

The circuit features **positive feedback** because any perturbation produces a response on the circuit that tries to increase it.

Observe that if the perturbation was negative, the same procedure will race down the V_o voltage until we reach the $V_{O\min}$ saturation voltage. We know the linear equilibrium point is not stable and we know that the operation will drift towards one of the saturation regions. But we cannot predict in which one of the two saturation regions we will end.

How does the first circuit, the good one that really follows, compares?

In this circuit V_i is connected to the (+) input and the amplifier output is connected to the (-) input.



A gentle push on V_o towards higher voltages will increase the differential $V_{(-)}$ voltage:

$$V_{(-)} = 1V + \Delta V \quad V_d = V_{dEQ} - \Delta V$$

That yields a voltage decrease in the output:

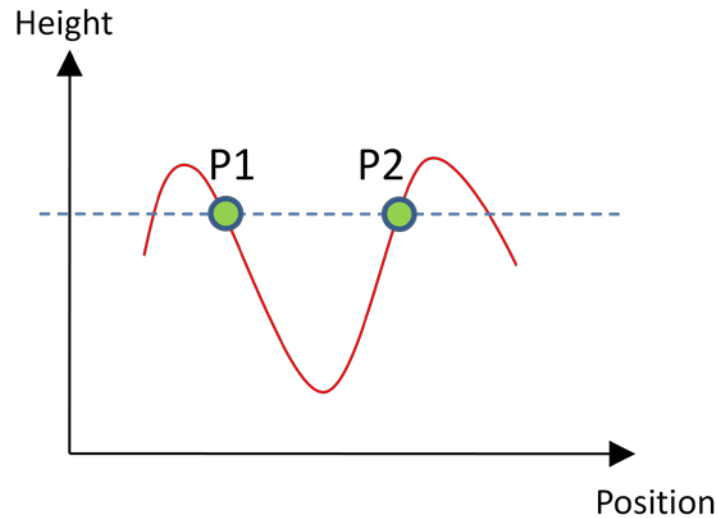
$$V_o = A(V_d) = A(V_{dEQ} - \Delta V) = A \cdot V_{dEQ} - A \cdot \Delta V = 1V - A \cdot \Delta V$$

The circuit features **negative feedback** because any perturbation produces a response on the circuit that tries to decrease it.

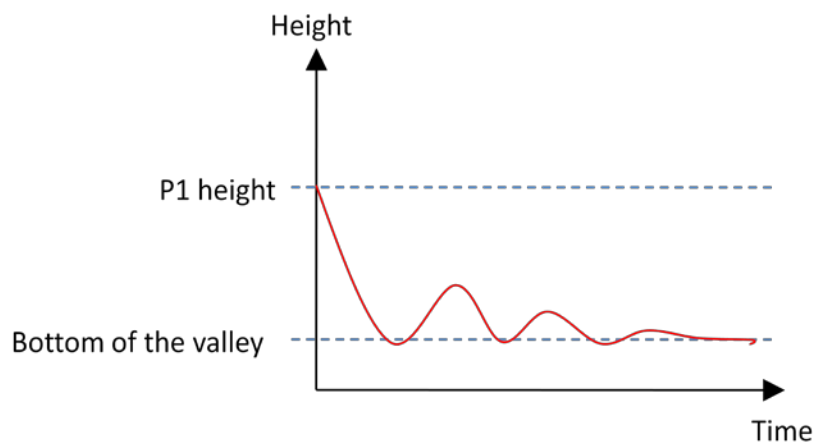
Negative feedback opamp circuits *usually* have a feedback connection from the output to the (-) input. But this is not always the case as one element on the feedback path could change the voltage polarity and require end up in (+). So, don't count on that cheap trick to always work.

Is negative feedback enough to have a stable system? Well, not exactly.

Returning to the roller coaster, imagine that we have the train still in the position P1 in the following figure.



If we let go the train, it will go down to the valley, but when it reaches it, it has gained enough speed to climb up the other side. It won't reach P2 because it has lost part of the energy due to friction. In the end the train will do several movements from one side of the valley to the other until it finally stops. We say that the system is **underdamped** because it oscillates around the equilibrium point before stopping.



We can be grateful of friction because it makes the train finally stop. If there was more friction, or a lower slope, the system could reach the bottom of the valley without any oscillation. In this case we say that the system is **overdamped**. The limit between this two cases is a **critically damped** system where we reach the equilibrium point as fast as possible without oscillations. If there was no friction at all with the rails or the air, the train will eternally move between points P1 and P2 making the system a mechanical oscillator.

If we try too hard to compensate the perturbations, as is the roller coaster case without friction, the system will oscillate without being able to rest on any equilibrium point.

Using negative feedback, as in the first follower, makes the circuit stable, but unity gain, as obtained in the follower, is just on the limit of stability for most opamps. This is due to the fact that the follower uses a lot of negative feedback. As $V_{(-)} = V_O$, the negative feedback is as big as the output perturbation. In general, any negative feedback circuit that feeds back only a fraction of the perturbation will also be stable. However, you can't guarantee the stability if you give a negative feedback to the input that is greater than the output perturbation.

So the general conclusions are:

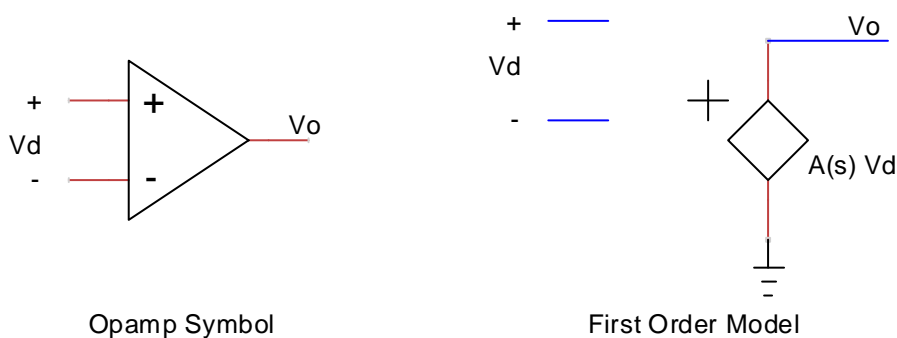
- **Positive feedback** : Circuit is not stable and will drift towards one of the saturation regions.
- **Some negative feedback**: Circuit is stable
- **Too much negative feedback**: Circuit will oscillate during some time if underdamped or continuously if not damped at all.

The ideal first order opamp model

This section deals with circuit analysis on the S domain. You need to have previous knowledge about circuit theory in the S domain to proceed in the rest of this document.

As we have seen, the stability of linear opamp circuits depends on the time response of the device. We addressed the stability of the circuit relating it to positive and negative feedback in a qualitative way.

In engineering it is important to grasp the qualitative concepts of circuits, but it is also important being able to give quantitative results. And, for that, we need better models.



The zero order basic model used in the previous project is enough to solve a linear circuit using the concept of virtual short circuit but is not able to give any result regarding its stability. A better opamp model models the time dependence of the amplifier output with a gain defined in the "s" domain $A(s)$.

The s domain relates to the [Laplace Transform](#). This transform converts a function defined in the time domain to a function defined in the “s” domain.

$$F(s) = \int_0^{\infty} f(t)e^{-s \cdot t} dt$$

The response of a circuit in the time domain can be transformed to an equivalent response in the “s” domain where some manipulations are easier to perform.

Opamps can be designed with different A(s) behaviors, but most opamps are designed in a way that a first order one pole model is enough for most calculations. Manufacturers design this kind of opamps by inserting a low frequency **dominant pole** that dominates the frequency response of the device.

A first order opamp model supposes that the amplifier gain defined as:

$$A(s) = \frac{A_o}{1 + \frac{s}{p_1}}$$

Where A_o is the DC gain (at zero frequency) and p₁ is a real pole in the negative region of the “s” plane. That is, the pole is found when s = -p₁. At the pole position the denominator is zero so A(s) goes to infinite. Hence the word “pole”.

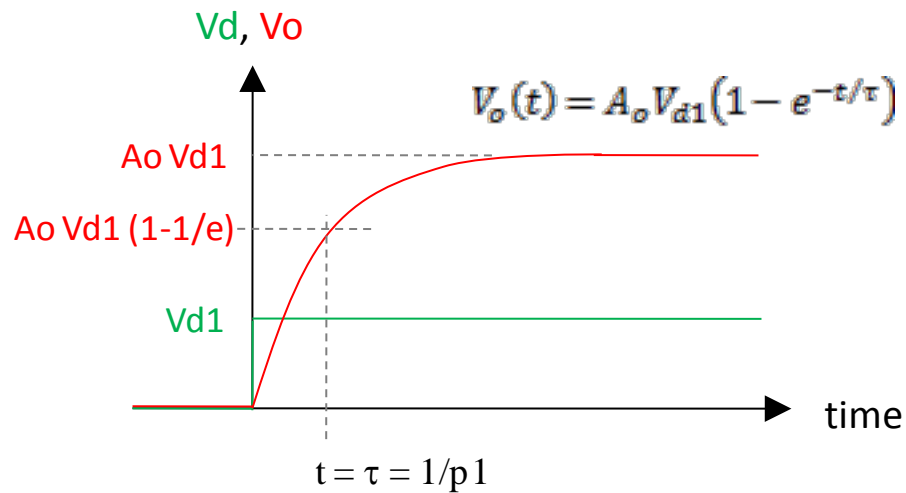
In this text, and the rest of the documents, we will call the negative region of the “s” plane to the region with negative **real** values. It is not useful to make differences in the positive and negative imaginary regions because poles (and zeros) with imaginary components come always in complementary pairs.

A pole is a point in the “s” plane that gives an unbound A(s) that goes to infinity. You can have one or several of them.

The time response of a dominant pole opamp against a step in V_d from 0 to V_{d1} can be calculated, for times greater than zero as:

$$V_o(t) = A_o V_{d1} (1 - e^{-t/\tau})$$

Graphically it will be:

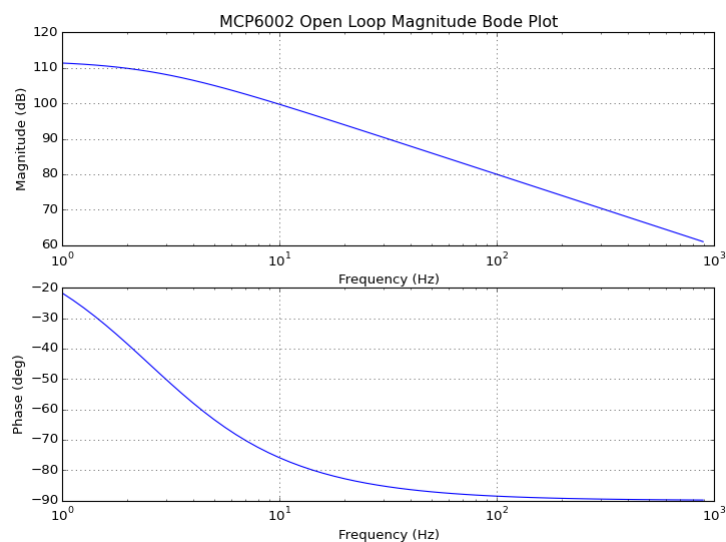


Note that the figure is not drawn with the same scale for V_d and V_o because A_o usually is greater than 1000.

We can compute the frequency domain response from the "s" plane $A(s)$ by setting $s = j\omega$, where ω is the angular frequency.

$$s = j\omega \quad \omega = 2\pi f$$

The following figure shows the open loop response of the MCP6002 opamp when using the dominant pole model.



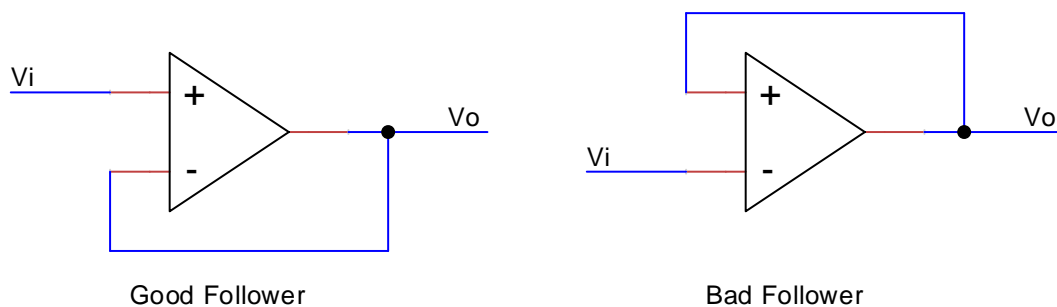
6

Compare the above response with the figure shown for the Open Loop Gain frequency response shown in the MCP6002 datasheet.

Frequency response of the follower circuit

Using the first order model, it is easy to check the stability of a circuit because a system, in order to be stable needs to have its poles in the left (negative) side of the "s" plane. We see, for instance, that the opamp, without any feedback is stable because it features one pole p_1 on the negative side of the "s" plane.

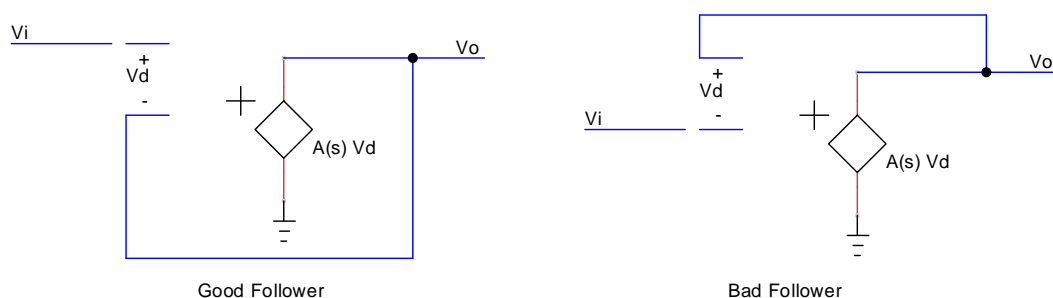
Let's analyze two circuits: The first, or "Good follower" is a correct voltage follower that features **negative feedback**. The second, or "Bad follower" is the **positive feedback** version of the proper follower circuit.



In order to obtain the response of the circuits, we only need to solve the transfer function $H(s)$ for the system:

$$H(s) = \frac{V_o}{V_i}(s)$$

In order to do that, we substitute the opamp symbol with the first order model as shown before and solve it to obtain the transfer function. During the calculations you can assume that A_o is much greater than 1.






7

Obtain the frequency response $H_1(s)$ for the good follower and $H_2(s)$ for the bad follower as function of A_o and p_1 .

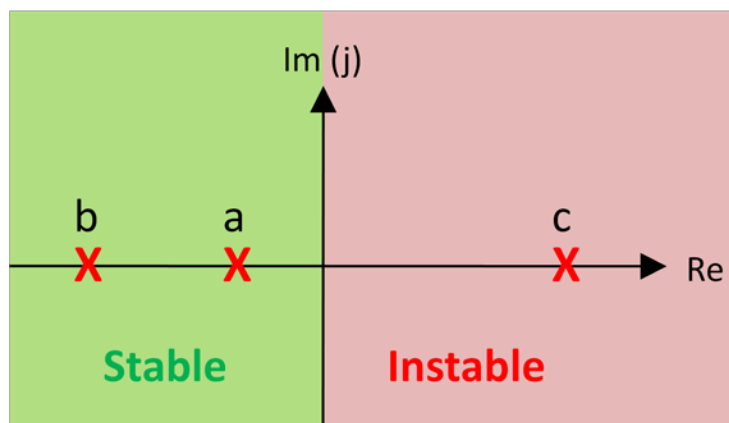
Simplify considering A_o much greater than 1.

Which are the zero frequency gain $H_1(0)$ and $H_2(0)$?

Where are the poles located in both cases?

If you have made the proper calculations you should find that the good follower has a real negative pole and that the bad follower has a real positive pole. As positive poles lead to instability, the result obtained in  is explained in a quantitative way.

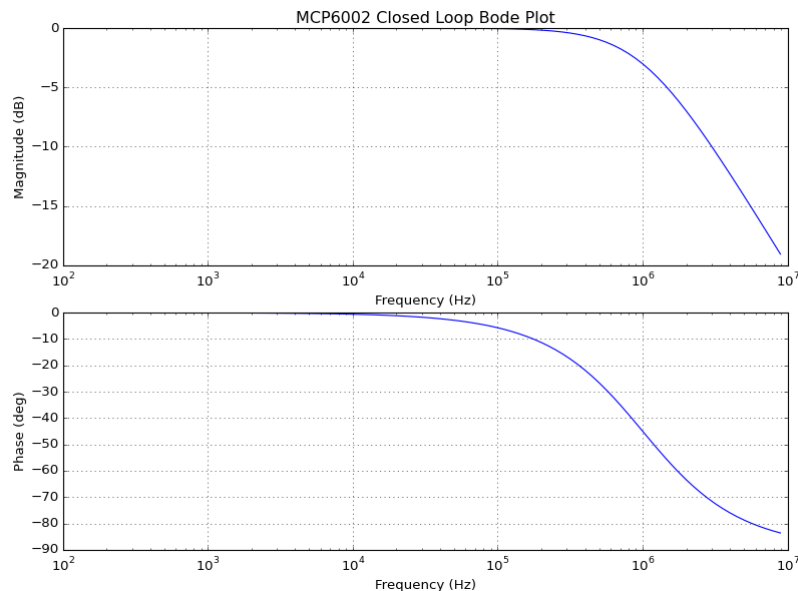
Note also that the negative pole on the good follower has a more negative value than the original system. It seems that the system should be more stable because we are deeper on the negative region of the "s" plane. In practice, as we will see, results in practice can be quite different. The following figure shows the pole locations we have calculated:



The original opamp had a pole in (a). The good follower has moved the pole to the (b) position and the bad follower has moved the pole to the (c) position. Feedback don't change the number of poles, just moves them around.

The Gain-Bandwidth product

Notice also that, as the pole absolute value has increased in the good follower, it has more bandwidth than the original amplifier. The frequency of the MCP6002 dominant pole was lower than 3 Hz whereas the follower has a much greater bandwidth.



The above plot shows the calculated magnitude bode plot for the MCP6002 opamp in follower configuration. Observe the decrease of gain and the increase of bandwidth compared with the open loop response.

You should see that the amplifier in open loop had a high gain A_o and a low bandwidth p_1 . The follower, instead, has a lower gain of just one (0 dB) and a high bandwidth equal to the $A_o \cdot p_1$ product. For a first order system we can define a Gain Bandwidth (GBW) product as the product of the gain absolute value and the bandwidth. In the case of the opamp it gives:

$$GBW = Gain \cdot Bandwidth = A_o \cdot p_1$$

It is interesting to see that the GBW is the same for the amplifier alone and for the follower circuit. It also can be demonstrated that the GBW is the same for any non time dependent negative feedback between those two options.

The GBW Gain Bandwidth Product for the MCP6002 can be obtained from the manufacture's datasheet:

Parameters	Sym	Min	Typ	Max	Units	Conditions
AC Response						
Gain Bandwidth Product	GBWP	—	1.0	—	MHz	

We have a typical GBW of 1MHz and, as we remember, a typical DC Ao gain of 112 dB. Opamp manufactures don't list the p1 value in the specifications. You need to deduce it from the Ao gain and GBW value.



8

Calculate the closed loop p1 frequency from the Ao gain and the GBW value.
Does the result agree with the open loop frequency response?

Important note: The GBW product invariance is a property of feedback in first order systems, not a general property of all linear systems. Moreover, it relates to the noise gain that not always is the same as the signal gain. Beware not to use the GBW product invariance out of its proper context.

Going over the edge

In this section we will wreak havoc on the opamp feedback.

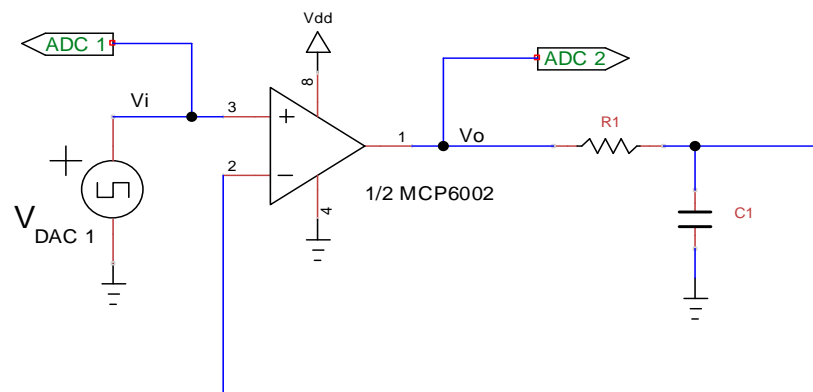
Opamps are the basic building blocks of linear analog circuits, but they depend on negative feedback to operate. Designing systems that operate on negative feedback is tricky. Opamps manufacturers want to ease the work of the circuit designers so that they buy more opamps from them. The end result is that most opamps are designed specifically so that they are easy to use on negative feedback systems. You normally don't need to think about feedback. It just works.

In general, a negative feedback circuit based on a normal opamp with a dominant pole will have good stability if we only use a **negative attenuation feedback** network that doesn't add significant **delays** to the feedback signal. The follower we had tested doesn't add any delay but it also doesn't give any attenuation. That's why for most opamps, the follower is the circuit that is nearer the edge of instability.

There are two easy things that you can do to make things hard for the opamp in a feedback system. Fortunately none of the typical opamp circuits feature those nasty things:

- Add delay to the feedback path
- Add gain to the feedback path

The following figure shows one example of the first method. We will drive this circuit with a 1V to 2V 100Hz square wave as in a previous case but we will add delay to the feedback.



The delay added to the feedback path depends on the R1 and C1 components. We can try the following 5 cases that add poles with time constants from 10 μ s to 3.3ms:

Case	τ	R1	C1
#1	10 μ s	10 k Ω	1 nF
#2	100 μ s	10 k Ω	10 nF
#3	330 μ s	33 k Ω	
#4	1 ms	100 k Ω	
#5	3.3 ms	330 k Ω	

If we have not closed the SLab system we should conserve the setting from the previous measurements:

```
>>> slab.waveSquare(1,2,100)
>>> slab.setWaveFrequency(100)
>>> slab.setTransientStorage(500,2)
```

So, we can perform the measurement just issuing at each case:

```
>>> slab.wavePlot(0)
```

9

Mount the circuit and check Vi and Vo on the oscilloscope for all cases from #1 to #5.


How is the system stability? Compare to the previous follower case without the R1 and C1 network.

When we change the output, an oscillation with decaying amplitudes until the stable voltage is reached is called **ringing**. A system with ringing is stable if it finally reaches a constant voltage, but too much oscillation in general is not good for the operation of the circuit.

Response of the two pole system

Crossing this line there are dragons

No kidding, the rest of this document ramps-up the complexity of the concepts.

In the circuit measured on  we saw that adding a delay in the feedback path using an RC circuit made the system less stable. We can analyze this system with the following equations:

$$Vd = Vi - V_{(-)} \quad Vo = A(s)Vd \quad V_{(-)} = Vo \cdot f(s) \quad f(s) = \frac{1}{1 + \frac{s}{p2}} \quad p2 = \frac{1}{R1 \cdot C1}$$

After some math we get:

$$H_3(s) = \frac{Vo}{Vi} = \frac{A(s)}{1 + f(s)A(s)}$$

In order to solve $H_3(s)$ we just need to substitute $A(s)$ and $f(s)$ in the previous expression.


$$A(s) = \frac{A_o}{1 + \frac{s}{p1}} \quad f(s) = \frac{1}{1 + \frac{s}{p2}}$$



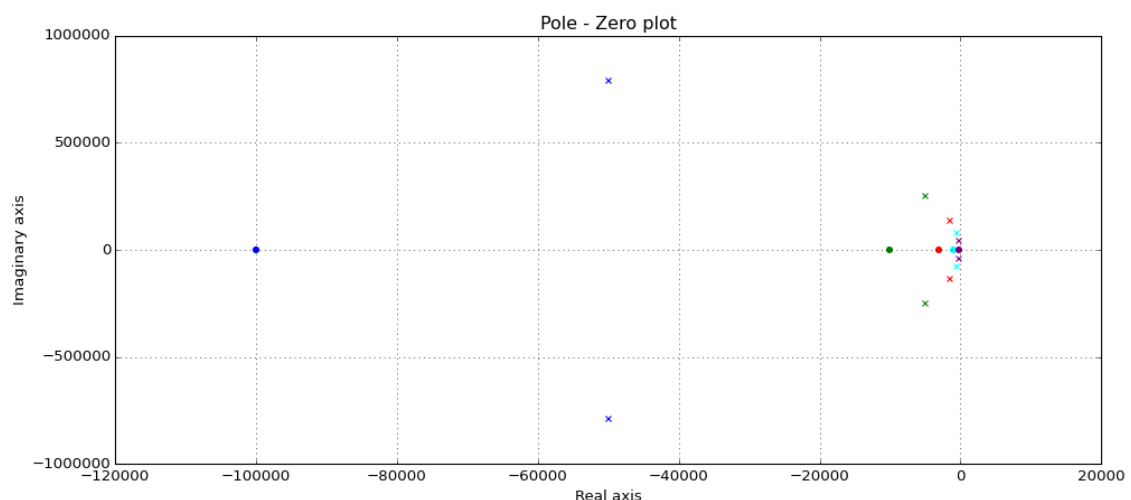
10

Obtain $H_3(s)$ as function of A_o , $p1$ and $p2$.


Locate how many poles and zeros are in the system.

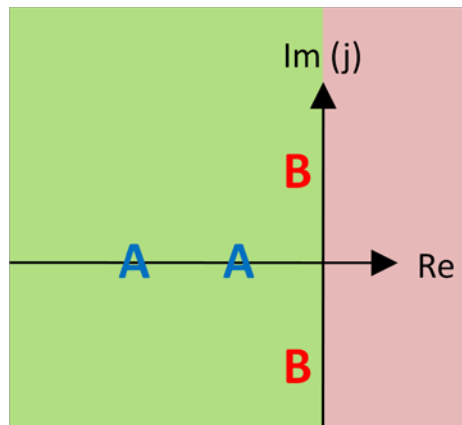
If you are brave enough, calculate the pole and zero positions for the 5 combinations of $R1$ and $C1$ we have tested before on .

The H_3 solutions feature a real zero and two complex poles, all of them in the negative region of the "s" plane. If you had been able to calculate pole and zero positions, you can check with the following diagram:



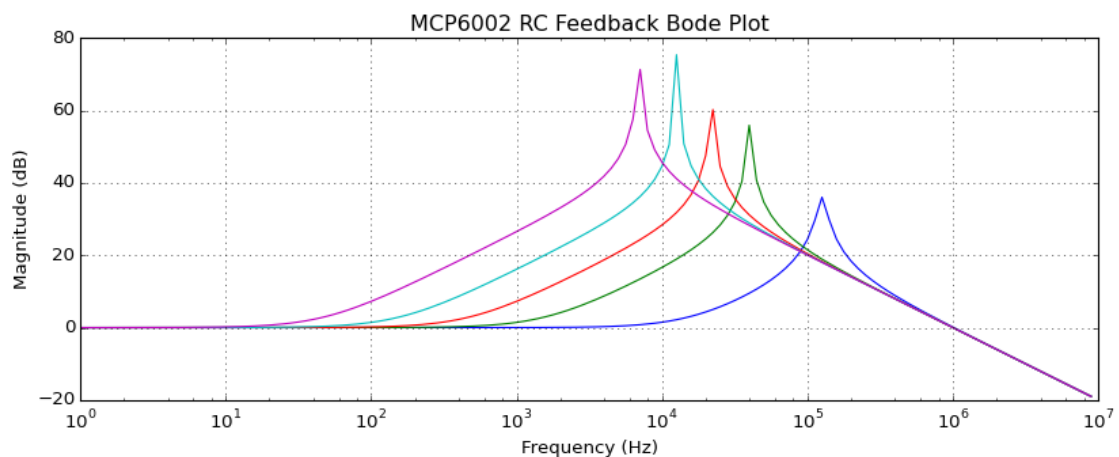
In the diagram, poles are marked with "x" and poles with "o". The five different colors correspond to the five possible time constants for the RC filter starting from blue for the 10 μ s case and ending at purple in the 3.3 ms case.

In the cases tested in , before connecting C1 to the (-) input, we had a two real pole system with poles p1 and p2, like the "A" poles in the following figure. One pole p1 was from the amplifier and another p2 from the R1 C1 feedback filter. As both poles are real, the system is [overdamped](#) and no oscillations are present.




After implementing the feedback the two poles can move to the "B" position as two imaginary poles that could be close to the positive instable "s" region. This system can be quite [underdamped](#) giving strong oscillations before reaching the final state as we have seen in the measurements.

The following figure shows the H₃ magnitude bode frequency response for the system for the five possible time constants in the RC network. Colors are the same as in the pole-zero diagram.



Don't rely on the peak value. As those bode diagrams are calculated using discrete points, there is no guarantee that we have calculated for the maximum. The worst case is the purple line for the 3.3 ms filter case as in this case gain is maximum at resonance.

In practice, no opamp features only one pole. There is always at least another pole up in the frequency range, usually beyond the rated GBW of the amplifier. That means that giving enough negative feedback it is usual to obtain a two pole underdamped system that will wildly oscillate. Due to that, that the negative real pole obtained for H_1 in 7 is not always realistic because real opamps feature more than one pole although the second pole is usually beyond the GBW product.

Last comments

Adding negative feedback to an amplifier is tricky because it can make it to oscillate. Opamp manufacturers use internal frequency compensation to guarantee that the opamp is stable up to the unity gain. This method, called dominant pole, drastically reduces the bandwidth of the amplifier but also makes it much easy to work with it.

As amplifier stability is an important subject in a feedback system, we will keep working on it on following SLab projects.

References

SLab Python References

Those are the reference documents for the SLab Python modules. They describe the commands that can be carried out after importing each module.

They should be available in the **SLab/Doc** folder.

TinyCad

Circuit images on this document have been drawn using the free software TinyCad

<https://sourceforge.net/projects/tinycad/>

SciPy

All the functions plots have been generated using the Matplotlib SciPy package.

<https://www.scipy.org/>

Copyright © Vicente Jiménez (2017)

This work is licensed under a Creative Common Attribution-ShareAlike 4.0 International license. This license is available at <http://creativecommons.org/licenses/by-sa/4.0/>

