SLiCAP: Symbolic Linear Circuit Analysis Program

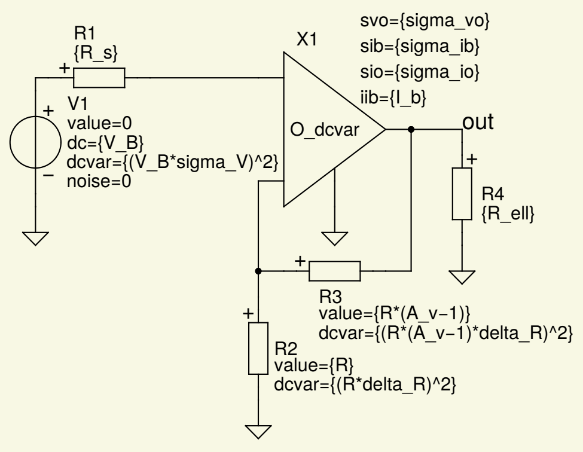
MATLAB® scripts for Analog Design Automation http://www.analog-electronics.eu

Symbolic and numerical evaluation of standard deviation of DC voltages and currents

Budgetting

1. Assign numerical values to known circuit elements
2. Obtain design equations for unknown parameters, such as:
   - Bias current tolerances
   - Offset voltages
   - Offset currents
   - Resistor tolerances
3. Obtain show stopper values by solving design equations with MATLAB
4. This yields search criteria for components

Circuit diagram



SLiCAP script

```
makeNetlist('dcBehavior', 'dcBehavior');
checkCircuit('dcBehavior');
source('V1');
detector('V_out');
simType('symbolic');
gainType('vi');
dataType('dcvar');
dcvarResults = execute();
```

DC network solution

$$\begin{pmatrix} V_1\\ V_2\\ V_3\\ V_{3X1}\\ V_{5X1}\\ V_{\mathrm{out}}\\ I_{V1}\\ I_{R2}\\ I_{R3}\\ \mathrm{Ii}_{F1X1}\\ I_{VoX1}\\ \mathrm{Io}_{N1X1}\end{pmatrix} = \begin{pmatrix} V_B - I_b\,R_s\\ V_B\\ V_B - I_b\,R_s\\ V_B - I_b\,R_s\\ 0\\ A_v\,V_B - I_b\,(R - R\,A_v) - I_b\,A_v\,R_s\\ -I_b\\ \frac{V_B}{R} - \frac{I_b\,R_s}{R}\\ \frac{I_b\,R_s}{R} - \frac{V_B}{R} - I_b\\ I_b\\ 0\\ \frac{I_b\,(R_s\,R_\ell + R\,A_v\,R_s)}{R\,R_\ell} - \frac{V_B\,(R_\ell + R\,A_v)}{R\,R_\ell} - \frac{I_b\,(R_\ell - R + R\,A_v)}{R_\ell}\end{pmatrix}$$

Variance DC detector voltage

$$\begin{aligned}&\sigma^2_{DET} =\\ &\sigma^2_{\mathrm{ib}}\,(R - R\,A_v + A_v\,R_s)^2\\ &+ {A_v}^2\,\sigma^2_{\mathrm{vo}}\\ &+ \sigma^2_{\mathrm{io}}\,(R\,A_v - R + A_v\,R_s)^2\\ &+ {A_v}^2\,{V_B}^2\,\sigma^2_V\\ &+ R^2\,{\delta_R}^2\,(A_v - 1)^2\left(I_b + \frac{V_B}{R} - \frac{I_b\,R_s}{R}\right)^2\\ &+ R^2\,\delta^2_R\,(A_v - 1)^2\left(\frac{V_B}{R} - \frac{I_b\,R_s}{R}\right)^2 [\mathrm{V}^2]\end{aligned}$$

Symbolic and numerical evaluation of source-referred and detector-referred noise

Budgetting

1. Assign numerical values to known circuit elements
2. Obtain design equations for unknown parameters, such as:
   - Input noise sources of operational amplifiers
   - Optimum geometry and operating current of CMOS JFET and BJT devices
   - Impedance of feedback networks
   - ...
3. Obtain show stopper values by solving design equations with MATLAB
4. This yields search criteria for components

Process noise results

Symbolic and numerical:
- Noise integration over frequency
- Correlated double sampling (CDS)
- CDS with integration
- Total RMS noise over frequency With / without CDS

Verification

1. Plot noise spectra versus frequency
2. Plot total RMS noise as a function of any circuit parameter

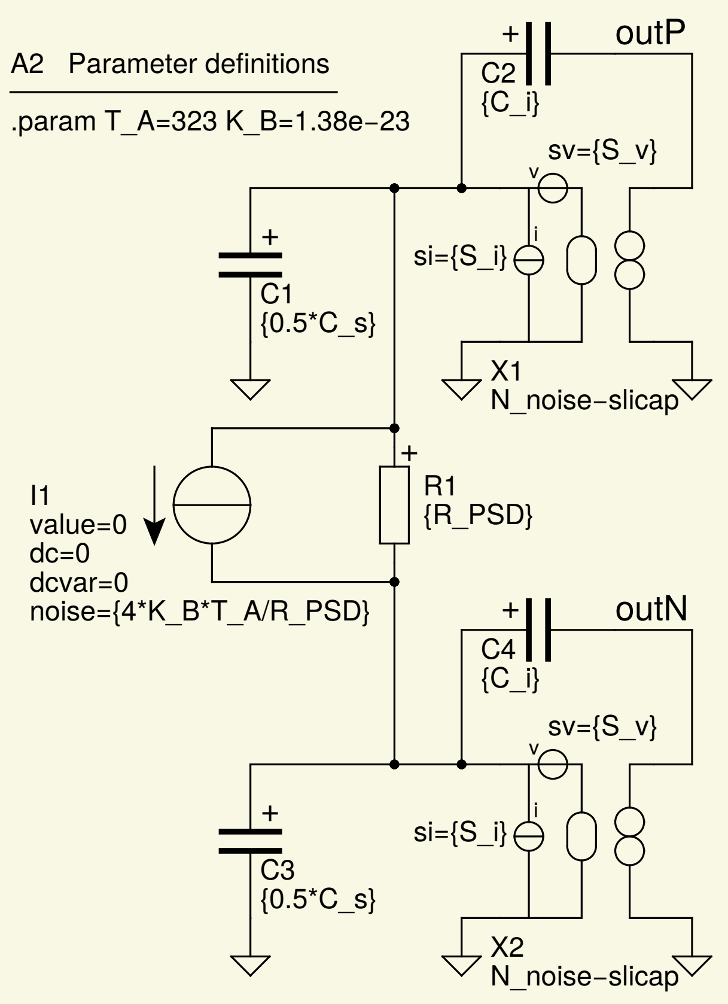
PSD pulse position detection system design of noise behavior

A1 Parameter definitions

.param C_i = 37.5p R_PSD = 64k C_s=20p

A2 Parameter definitions

.param T_A=323 K_B=1.38e−23



Design Task Determine show-stopper values for S_v and S_i such that the total differential RMS output noise over a bandwidth of 450kHz, after CDS with $\tau = 5\mu$s, is less than 50μV.

SLiCAP script

```
syms 'F' 'S_v' 'S_i';
F_min = 1;     % Minimum frequency
F_max = 450e3; % maximum frequency
vn    = 50e-6; % RMS diff output noise after CDS
tau   = 5e-6;  % CDS delay time
checkCircuit('PSDnoiseDesign');
simType('numeric');
gainType('vi');
dataType('noise');
detector('V_outN', 'V_outP');
onoise = getOnoise(doCDS(execute(), tau));
rmsSv = RMSnoise(subs(onoise, S_i, 0), F_min, F_max);
Sv_max = double(solve(rmsSv - S_v, vn));
rmsSi = RMSnoise(subs(onoise, S_v, 0), F_min, F_max);
Si_max = double(solve(rmsSi - S_i, vn));
disp(strcat('Sv_max =', sprintf('%9.2e', Sv_max), ' V^2/Hz'));
disp(strcat('Si_max =', sprintf('%9.2e', Si_max), ' A^2/Hz'));
```

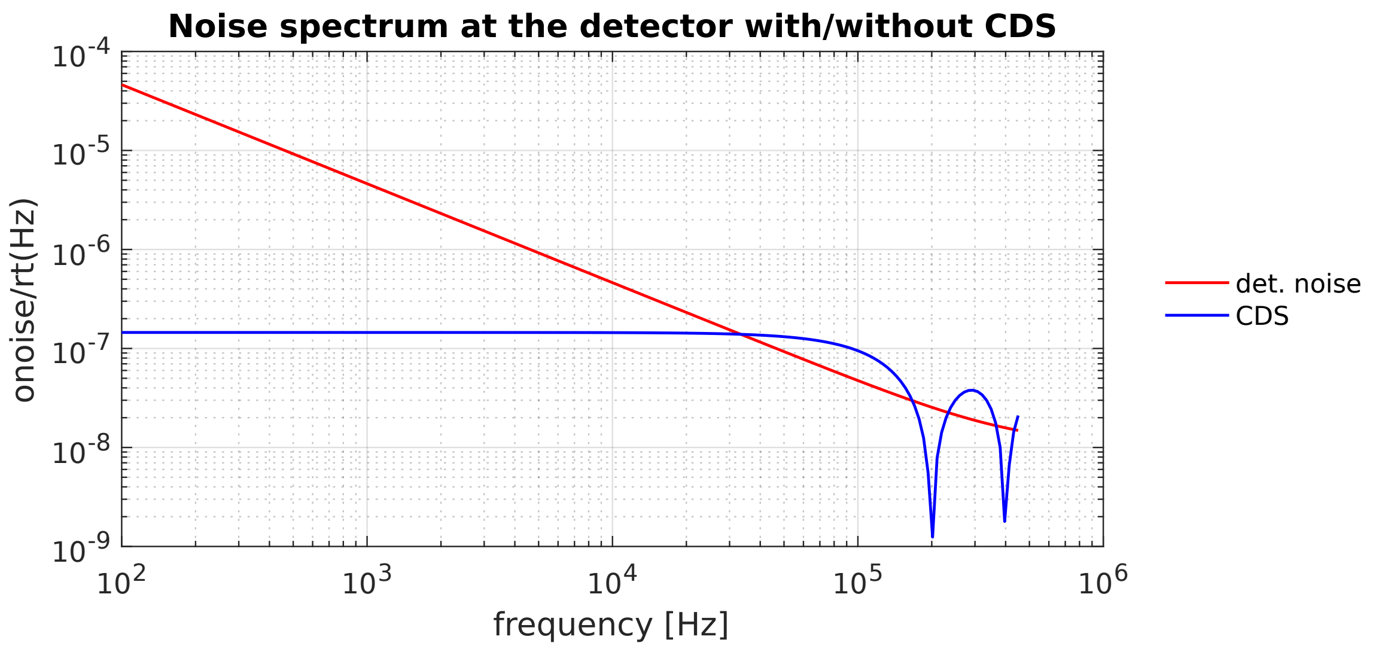
Result

```
Sv_max = 1.03e-16 V^2/Hz
Si_max = 1.82e-25 A^2/Hz
```

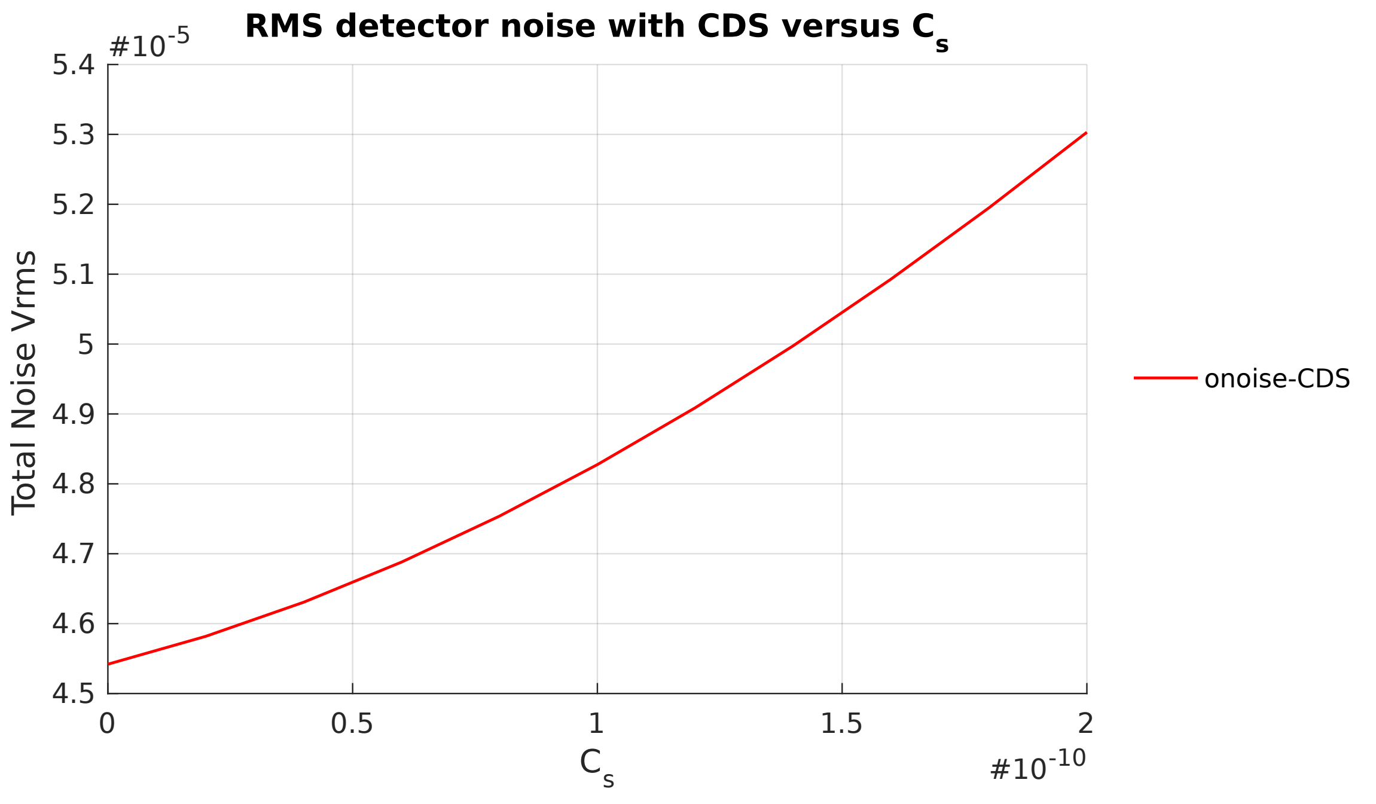
Select opAmp with

```
S_v = 6 nV/rtHz
S_i = 5 fA/rtHz
```

Noise spectrum at the detector with/without CDS



RMS detector noise with CDS versus C_s



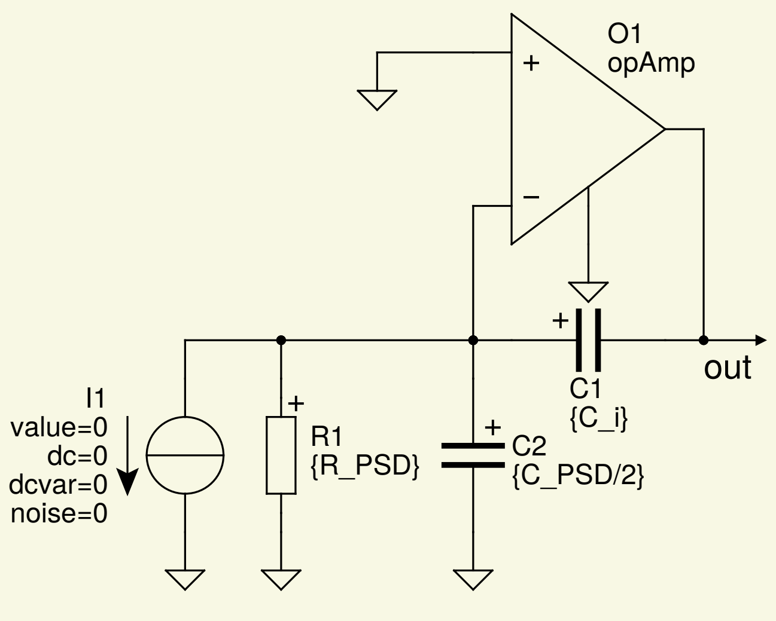
Symbolic and numerical evaluation of voltages and currents and of transfer functions of the asymptotic-gain feedback model

- Detector voltage / current
- Gain
- Asymptotic-gain
- Loop gain
- Servo function
- Direct transfer

Budgetting

1. Assign numerical values to known circuit elements
2. Obtain design equations for unknown parameters, such as:
   - DC gain of OpAmp
   - Gain-bandwidth product of OpAmp
   - Input capacitance of OpAmp
   - Output resistance of OpAmp
   - ...
3. Obtain show stopper values by solving design equations with MATLAB
4. This yields search criteria for the operational amplifier

PSD pulse position detection system design of dynamic behavior



A1 Parameter definitions

.param R_PSD=65k C_PSD=20p C_i=37.5p

A2 Model definition

.model opAmp OV av={A_0/(1+s*tau_1)} zo={R_o} cd={c_d}

Frequency compensation and design verification

1. Calculate servo bandwidth from loop gain rational expression
2. Plot magnitude, phase and/or delay versus frequency
3. Plot unit-impulse and/or unit-step responses
4. Plot root-locus with any circuit parameter as root-locus variable
5. Plot Nyquist plot of loop gain
6. Calculate Routh array
7. Adjust coefficients of gain rational expression with frequency compensation elements

Generate beautifully typeset HTML project documentation

1. MathJax typesetting of math
2. Include:
   a. Schematics and graphics
   b. MATLAB plots
   c. Equations and expressions
   d. CSV tables including LaTeX
   e. Text files including HTML and LaTeX
   f. Netlists
   g. Matlab script files
   h. Log messages
3. Sphinx compatibe RST files with search and page navigation menu
4. One-click update of project documentation with all expressions, graphs, tables, etc.

Select operational amplifier

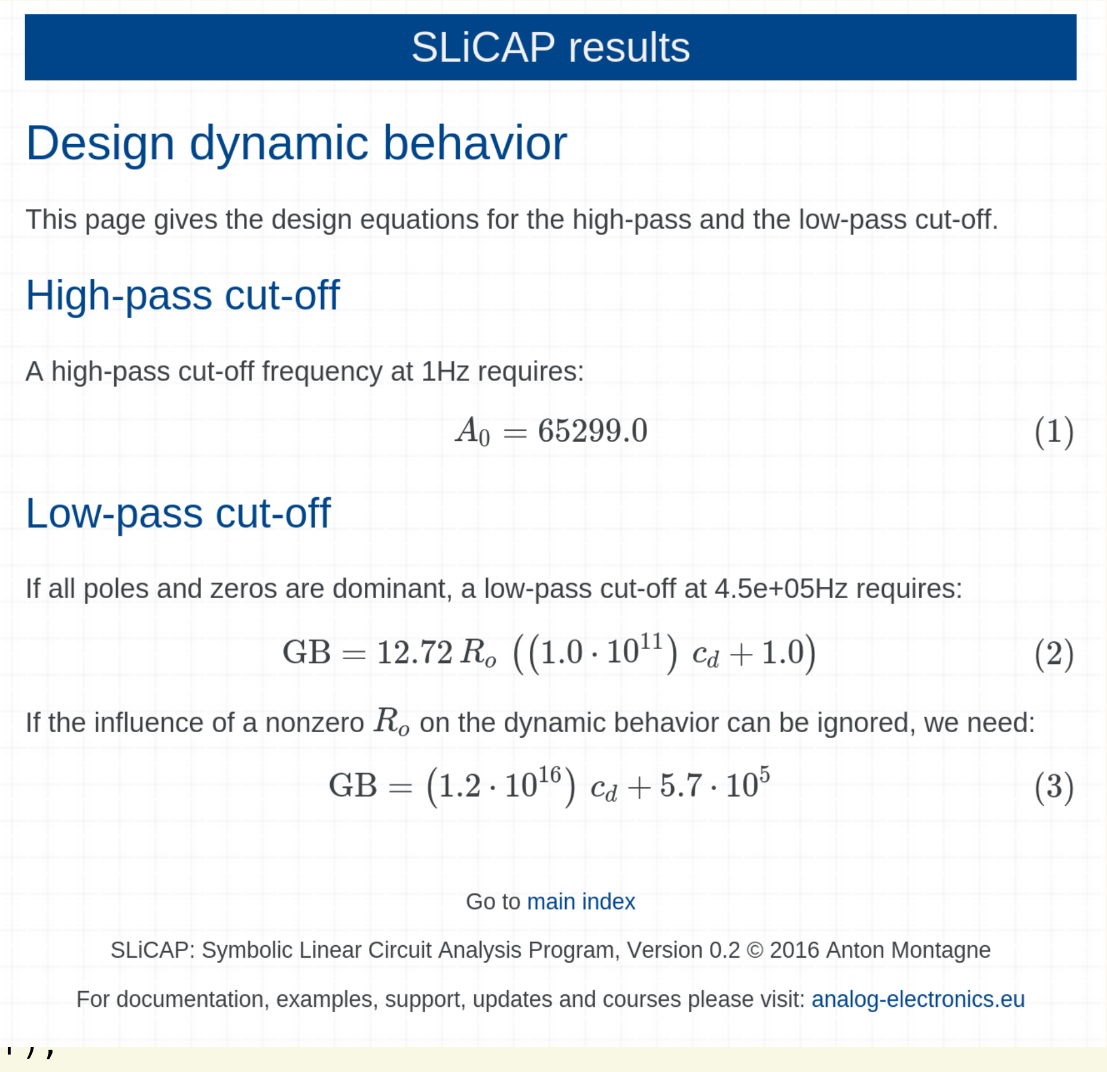
```
* file: AD8610.lib SLiCAP model for AD8610
.model AD8610 OV
+ cd = 15p
+ cc = 8p
+ av = {300k*(1-s/2/PI/120M)/(1+s*300k/2/PI/25M)/(1+s/2/PI/120M)}
+ zo = 20
```

Design Task

Determine show-stopper values for A_0, τ_1, c_d and R_o of the operational amplifier such that the integrator has a low-pass cut-off frequency above 450kHz and a high-pass cut-off frequency below 1Hz.

$GB = \frac{A_0}{2\pi\tau_1}$

```
makeNetlist('PSDlowPassHighPass', 'PSD dynamic behavior');
checkCircuit('PSDlowPassHighPass');
f_hp = 1;
f_lp = 450e3;
source('I1');
detector('V_out');
lgRef('E_O1');
simType('numeric');
gainType('loopgain');
dataType('laplace');
L = execute();
L.rational = L.results(1);
L.coeffs = coeffsTransfer(L.rational);
L.numerCoeffs = L.coeffs(1);
L.denomCoeffs = L.coeffs(2);
L.numerOrder = length(L.numerCoeffs)-1;
L.denomOrder = length(L.denomCoeffs)-1;
% If all loop gain zeros and no poles below servo high-pass cut-off:
syms 'A_0' 'c_d' 'R_o' 'tau_1' 'GB';
assume(A_0 > 0 );
assume(c_d > 0 );
assume(R_o > 0 );
assume(tau_1 > 0 );
Servo_fHighPass = (1/2/pi) * abs(L.numerCoeffs(L.numerOrder+1))^-(1/L.numerOrde
highPassCondition = solve(Servo_fHighPass-f_hp, A_0);
% If all poles and zeros below low-pass cut-off:
Servo_fLowPass = (1/2/pi) * abs(L.numerCoeffs(L.numerOrder+1)/L.denomCoeffs(L.denomOrder+1))^-(1/(L.numerOrder-L.denomOrder));
Servo_fLowPass = subs(Servo_fLowPass, tau_1, A_0/2/pi/GB);
lowPassCondition_Ro_cd = solve(Servo_fLowPass-f_lp, GB);
% If pole due to nonzero output impedance is not dominant:
L.rational = subs(L.rational, R_o, 0);
L.coeffs = coeffsTransfer(L.rational);
L.numerCoeffs = L.coeffs(1);
L.denomCoeffs = L.coeffs(2);
L.numerOrder = length(L.numerCoeffs)-1;
L.denomOrder = length(L.denomCoeffs)-1;
Servo_fLowPass = (1/2/pi) * abs(L.numerCoeffs(L.numerOrder+1)/L.denomCoeffs(L.denomOrder+1))^-(1/(L.numerOrder-L.denomOrder));
Servo_fLowPass = subs(Servo_fLowPass, tau_1, A_0/2/pi/GB);
lowPassCondition_cd = solve(Servo_fLowPass-f_lp, GB);
htmlPage('Design dynamic behavior');
text2html('This page gives the design equations for the high-pass and the low-pass cut-off.');
head2html('High-pass cut-off');
text2html(strcat('A high-pass cut-off frequency at', sprintf('%3d', f_hp), 'Hz requires:'));
eqn2html(A_0, highPassCondition);
head2html('Low-pass cut-off');
text2html(strcat('If all poles and zeros are dominant, a low-pass cut-off at', sprintf('%9.1e', f_lp), 'Hz requires:'));
eqn2html(GB, lowPassCondition_Ro_cd);
text2html('If the influence of a nonzero $R_o$ on the dynamic behavior can be ignored, we need:');
eqn2html(GB, lowPassCondition_cd);
stophtml();
```



SLiCAP results

Design dynamic behavior

This page gives the design equations for the high-pass and the low-pass cut-off.

High-pass cut-off

A high-pass cut-off frequency at 1Hz requires:

$$A_0 = 65299.0 \tag{1}$$

Low-pass cut-off

If all poles and zeros are dominant, a low-pass cut-off at 4.5e+05Hz requires:

$$\mathrm{GB} = 12.72\,R_o\,\left(\left(1.0\cdot 10^{11}\right)\,c_d + 1.0\right) \tag{2}$$

If the influence of a nonzero R_o on the dynamic behavior can be ignored, we need:

$$\mathrm{GB} = \left(1.2\cdot 10^{16}\right)\,c_d + 5.7\cdot 10^{5} \tag{3}$$

Go to main index

SLiCAP: Symbolic Linear Circuit Analysis Program, Version 0.2 © 2016 Anton Montagne

For documentation, examples, support, updates and courses please visit: analog-electronics.eu

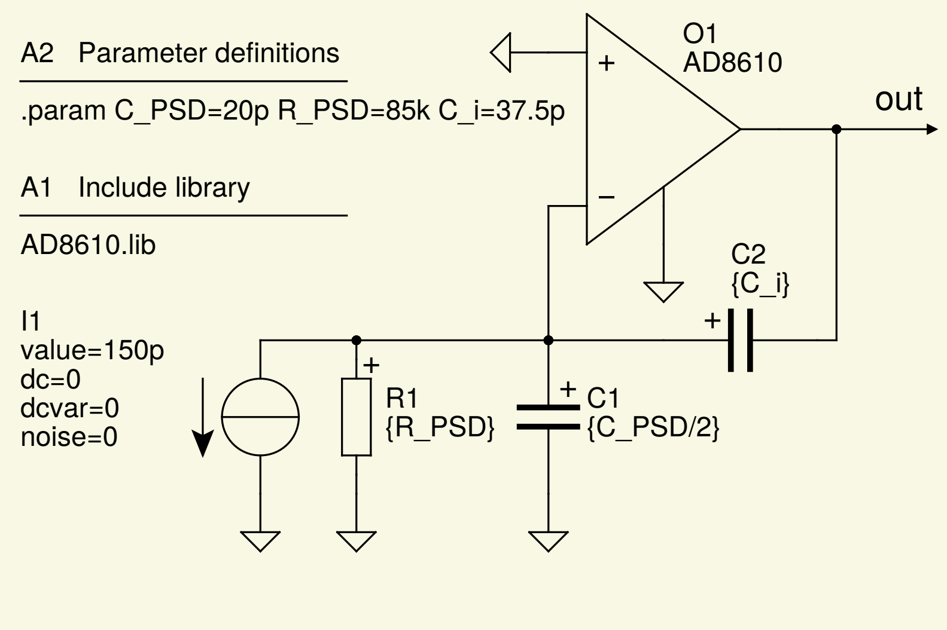
Integrator circuit

A2 Parameter definitions

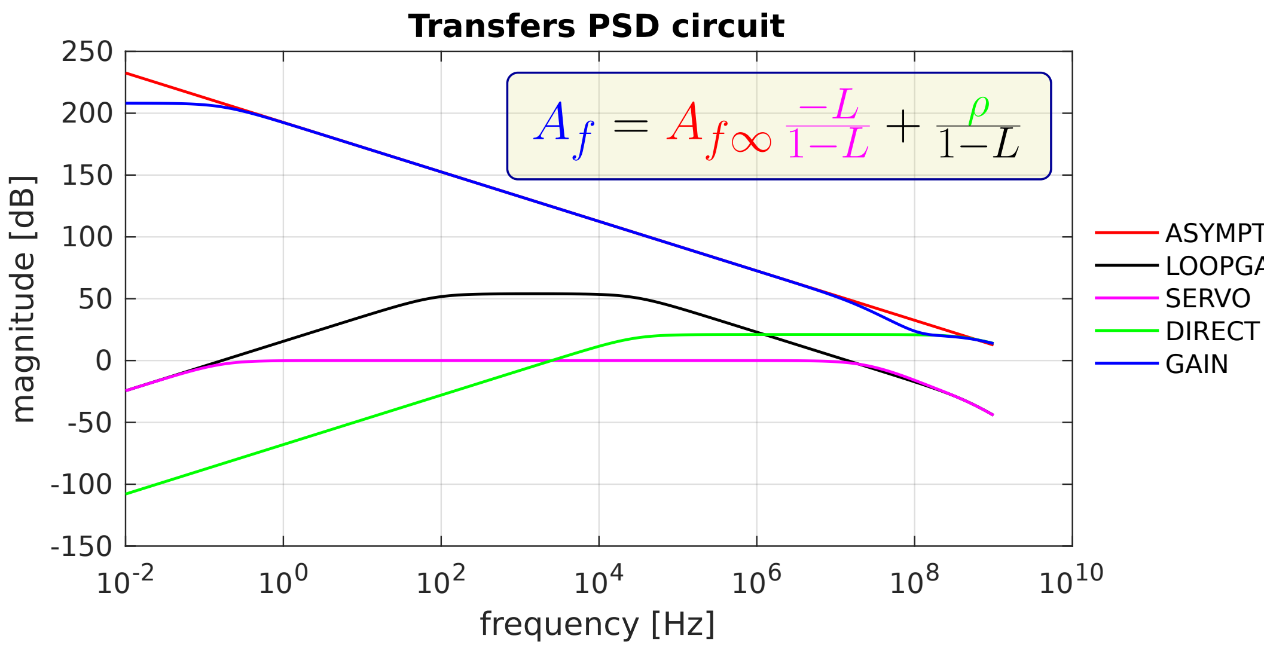
.param C_PSD=20p R_PSD=85k C_i=37.5p

A1 Include library

AD8610.lib



Transfers PSD circuit

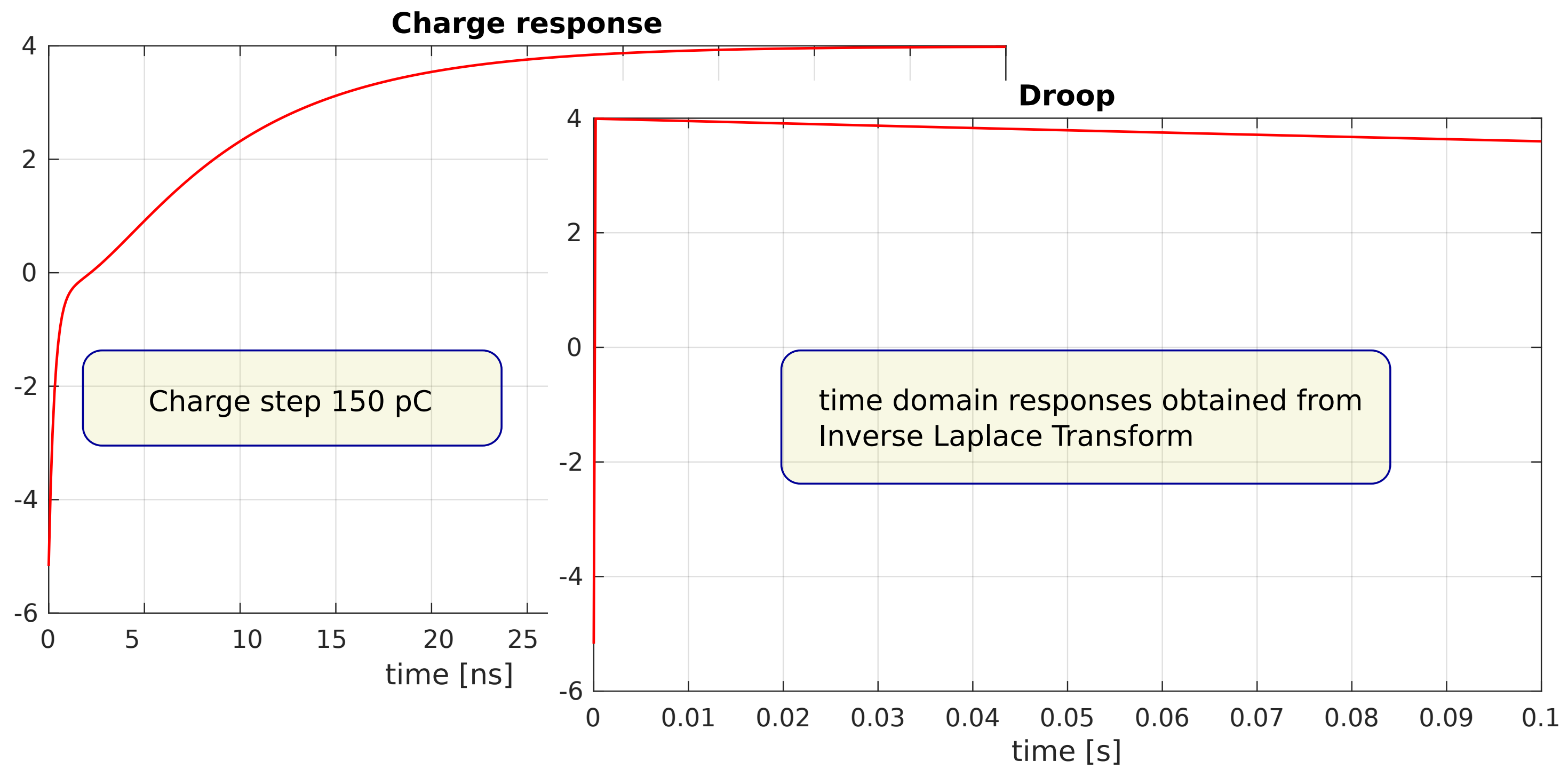


$$A_f = A_{f\infty}\frac{-L}{1-L} + \frac{\rho}{1-L}$$

```
findServoBandwidth(L.rational)
Servo high-pass cut-off at: 1.7e-01 Hz, order: -1
Servo low-pass cut-off at: 1.4e+01 MHz, order:  1
```

Charge response

Droop



Charge step 150 pC

time domain responses obtained from Inverse Laplace Transform