SPICE-Simulation using LTspice IV

Tutorial for successful simulation of electronic circuits with the free full version of LTspice IV (before named “SwitcherCAD”), available at Linear Technologies (www.linear.com).

Version 1.3

Copyright by
Gunthard Kraus, Elektronikschule Tettnang, Germany,
Guest Lecturer at the DHBW (Duale Hochschule Baden-Wuerttemberg),
Friedrichshafen, Germany

Email: krausg@elektronikschule.de or Gunthard.Kraus@gmx.de

Homepage: www.elektronikschule.de/~krausg

December 15th, 2010

Contents **Page**

1. Introduction

Modern electronics needs circuit simulation -- only in this manner you can save time, cost and effort when designing new or modifying existing circuits. Every new idea can be tested without a real printed circuit board or a soldering iron. Therefore the “**SPICE**” program was developed before 1980 at the Berkeley University, running on FORTRAN machines. But the more important version for the „normal“ user is **PSPICE** (= SPICE for the PC).

Around the „SPICE Kernel“ lot of people have programmed shells and programs for simple and intuitive usage and so you can find lot of software offers on the market. Most of the available programs are excellent, but huge and expensive and so we have to say „Thank You“ to Linear Technologies. They offer a free full SPICE-program named **„LTspice“** without any restrictions. It was foreseen to simulate switching power supplies using the semiconductors of the enterprise…..but can also be used for nearly other electronic purpose. It can be downloaded from the web without any problems or fees but the usage is a little tricky -- a mixture of command lines, menues and mouse clicks. So it needs a lot of effort before the first own simulation. And this was the reason for an old professor like me to write this tutorial for other people.

Warning for the new user:

There are sometimes several ways to get to the same command (hot keys, menu bar symbols, menu selection, menus that appear with a right mouse click, etc.). So find out the way which you personally prefer and don’t be afraid of or confused by the different kinds of run commands.

----

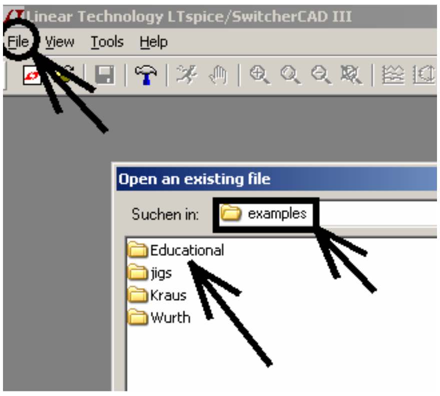
2. Installation

No problem: download the zipped software from the homepage (www.linear.com), click on the „.exe“-file and you will be guided by the program. An icon is created at last on your Windows-screen.

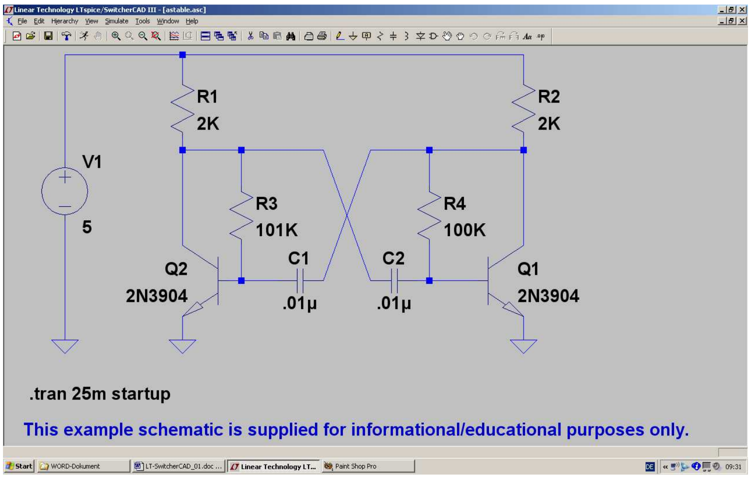
----

3. Exercises with the included Example „Astable Multivibrator“

3.1. Using Circuit and Simulation



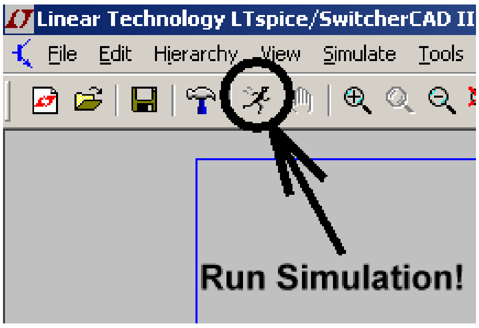
Click on „**File**“ in the upper left corner and then open „**examples**“. In the folder “educational” you find the file “astable.asc”. Open it.



.tran 25m startup

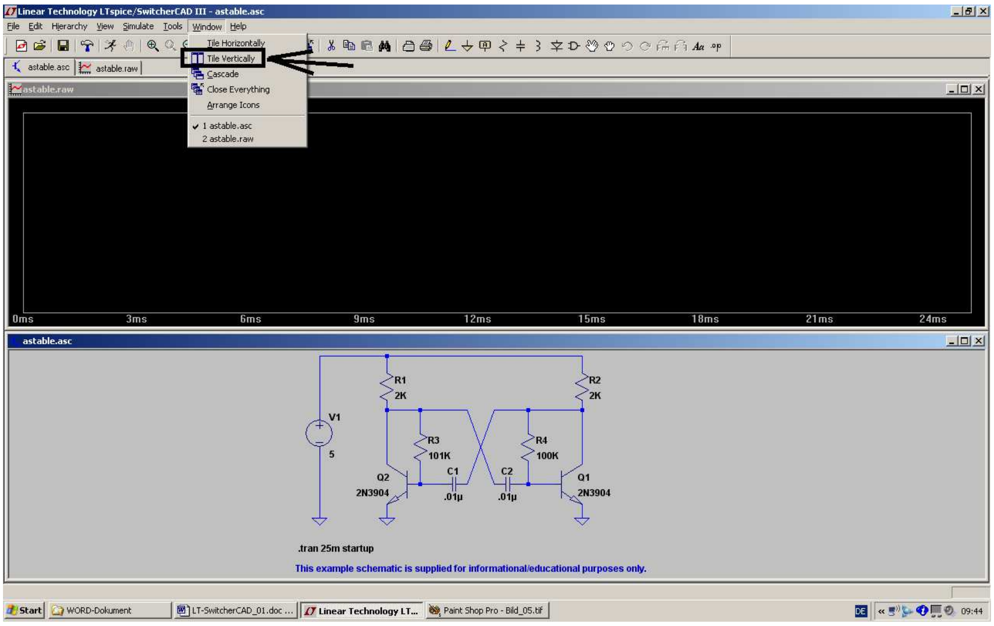
This example schematic is supplied for informational/educational purposes only.

So the screen with the schematic should now look like.


Linear Technology LTspice/SwitcherCAD II
File Edit Hierarchy View Simulate Tools

Run Simulation!

Now look for the button with the „running guy“. This starts the simulation.

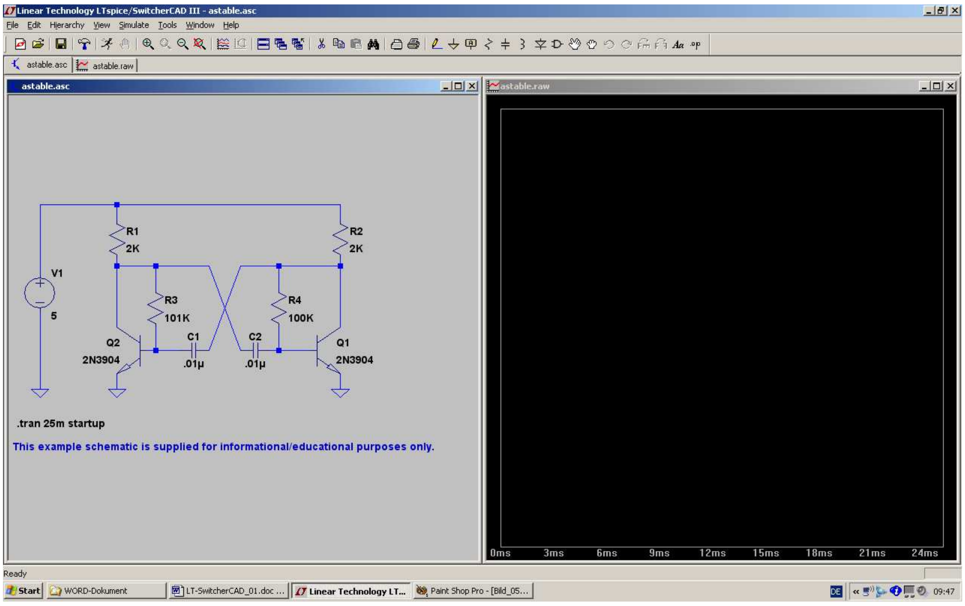


.tran 25m startup

This example schematic is supplied for informational/educational purposes only.

The simulation is done, but where are the results? (The diagram in the upper half of the screen is still empty....)

If you prefer another manner of presentation, open the menu „Window“ and click on „**Tile vertically**“.


Linear Technology LTspice/SwitcherCAD III - astable.asc
File Edit Hierarchy View Simulate Tools Window Help
astable.asc
astable.raw
R1
2K
R2
2K
V1
5
R3
101K
R4
100K
C1
.01µ
C2
.01µ
Q2
2N3904
Q1
2N3904
0ms
3ms
6ms
9ms
12ms
15ms
18ms
21ms
24ms
Ready
Start
WORD-Dokument
LT-SwitcherCAD_01.doc ...
Linear Technology LT...
Paint Shop Pro - [Bild_05...
09:47

.tran 25m startup

This example schematic is supplied for informational/educational purposes only.

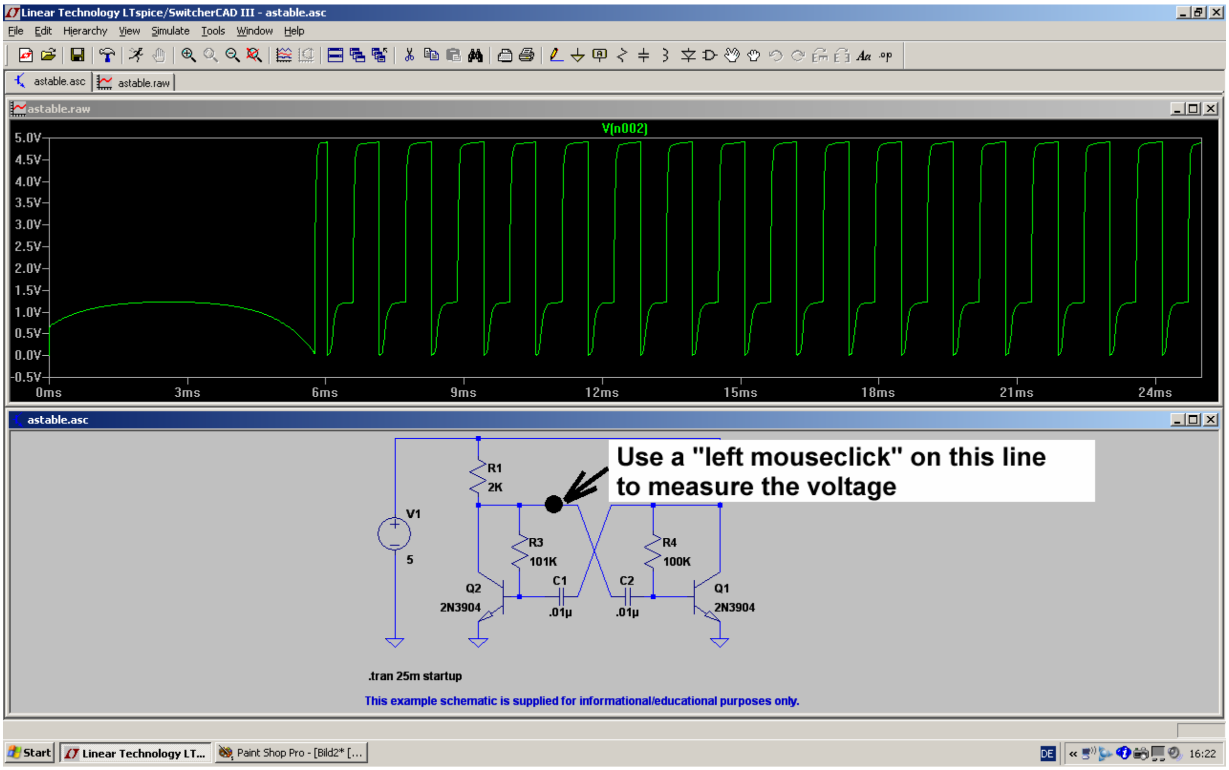
Better now for you?

Information:

Who likes a result plot with thick lines for the curves must now click on the “button with the hammer”. There he finds a menu ”Waveforms” and in it “Plot data with thick lines”.

3.2. Presentation of Simulation Results

Move the cursor to the point or line in the schematic of which you want to know the voltage. When arriving the cursor changes its form to a „testclip“. Now a simple mouseclick activates the presentation of the desired curve:

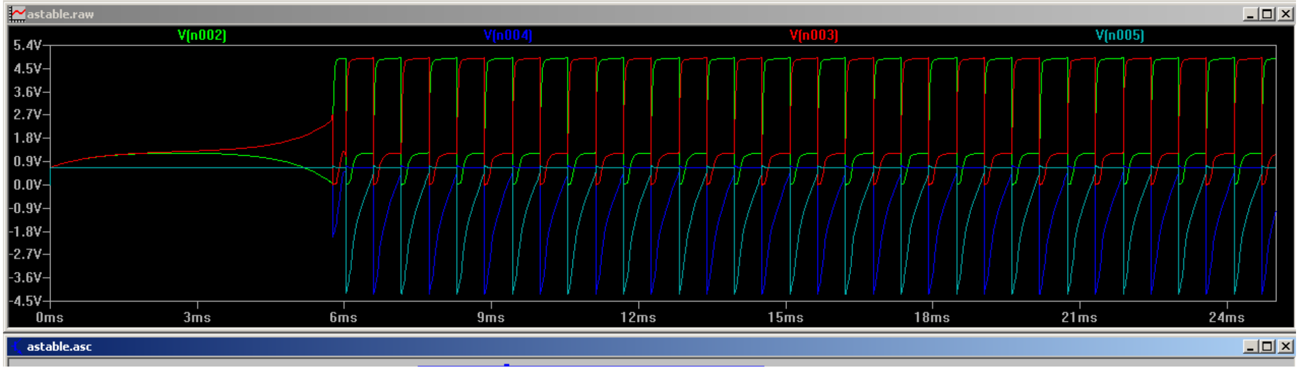


Use a "left mouseclick" on this line to measure the voltage

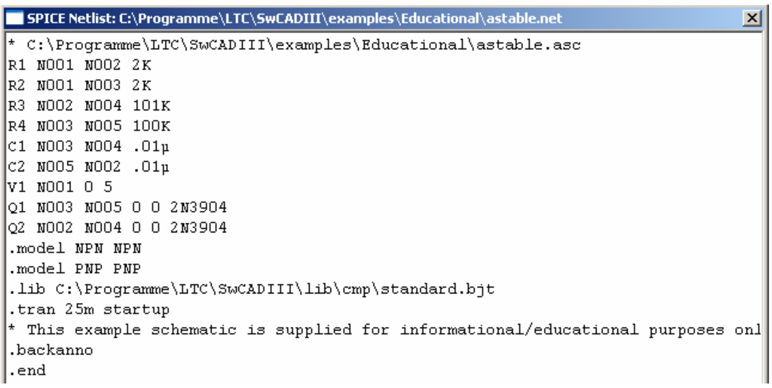
.tran 25m startup

This example schematic is supplied for informational/educational purposes only.

You can continue in this manner, but remember: you have only one diagram. And how is it possible to separate the different curves of the different points? Where can you get the informations from? For instance so the presentation of all base- and collector-voltages looks like:



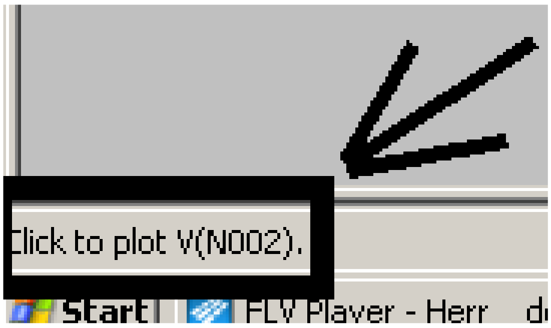
In principle you have a choice of 3 possibilities:


SPICE Netlist: C:\Programme\LTC\SwCADIII\examples\Educational\astable.net

```
* C:\Programme\LTC\SwCADIII\examples\Educational\astable.asc
R1 N001 N002 2K
R2 N001 N003 2K
R3 N002 N004 101K
R4 N003 N005 100K
C1 N003 N004 .01µ
C2 N005 N002 .01µ
V1 N001 0 5
Q1 N003 N005 0 0 2N3904
Q2 N002 N004 0 0 2N3904
.model NPN NPN
.model PNP PNP
.lib C:\Programme\LTC\SwCADIII\lib\cmp\standard.bjt
.tran 25m startup
* This example schematic is supplied for informational/educational purposes onl
.backanno
.end
```

a) Look for the node’s name in the „***Netlist***“, because the different curves are named “**V(node**”). You find this important list in the menu “**VIEW**” and „**SPICE Netlist**“.

This is the netlist for our example and easy to read. Every line belongs to a part in the schematic and begins with the part’s name, followed by the nodes between the part is inserted. At last you find the part’s value.

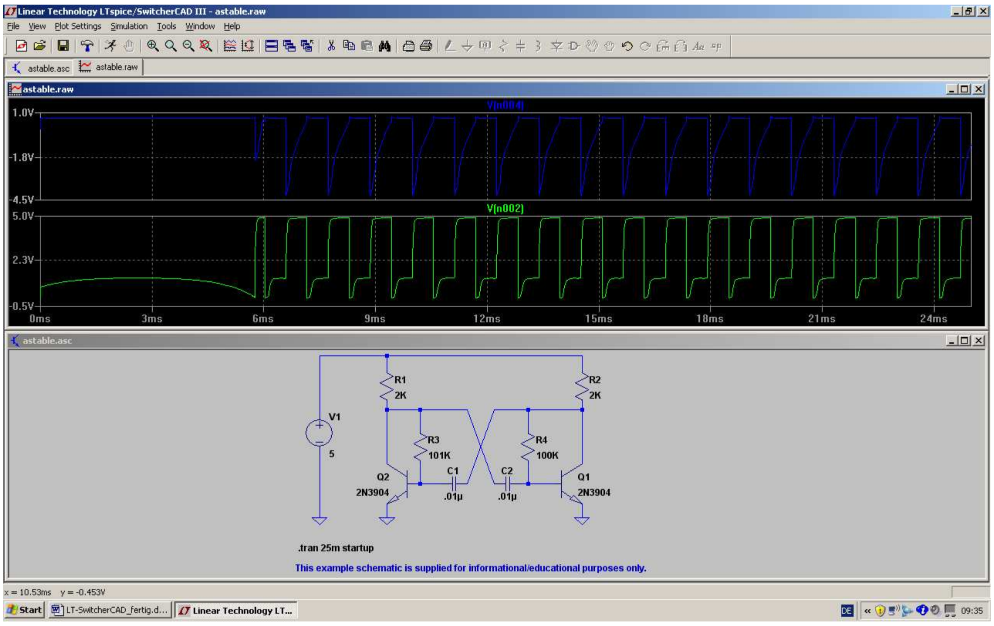


lick to plot V(N002).

b) Much easier is the cursor method. Move the cursor to the desired point and look for the cursor's change to a testclip.

In this moment you get a message in the lower left corner of the screen with the node's name

----



.tran 25m startup

This example schematic is supplied for informational/educational purposes only.

x = 10.53ms y = -0.453V

c)
Much easier is to use separate diagrams for the different curves.

Right click right on the presentation window and choose

„**Add Plot Plane**".

Now you get an additional diagram. Left click on it with the mouse to activate it. Afterwards click on the point in the schematic (f. i. the base of the left transistor) to present the desired voltage
If necessary, repeat this procedure to get more diagrams.

But how to delete undesired presented curves: see the following chapter!

3.3. Deleting Curves

This is important to know if you present too much curves in only one diagram and you are loosing the overview. So

a) Left click on the diagram to activate it -- the frame colour should change to a dark blue.

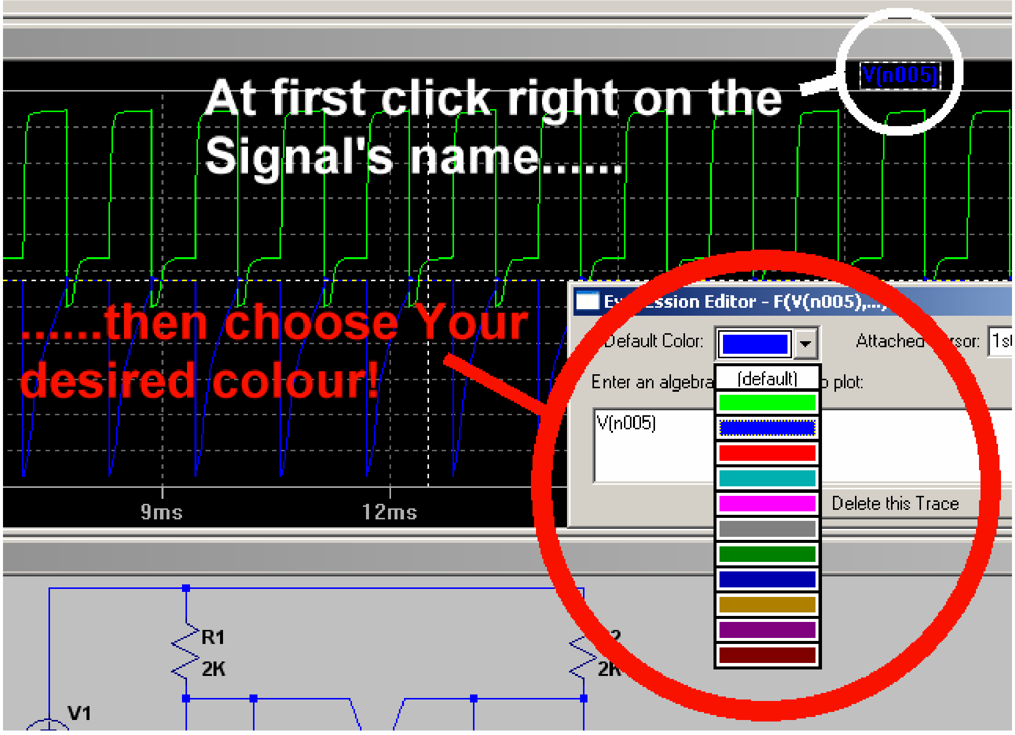
b) Press **F5** and the cursor will change to scissors.

c) Move the scissors to the name of the undesired curve -- e. g. "V(n001)" above the diagram and click on it. The curve will disappear at once.

Important:
To measure another voltage in the schematic, please press at first „**Escape**" on the keyboard to leave the delete function. Then click left on the schematic and watch that its frame changes to dark blue. If this is true, move the cursor to the desired point in the schematic until the form changes to a test clip…and so on....

3.4. Changing Curve Colours

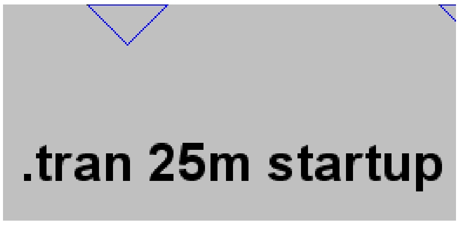
A very simple exercise.......



At first click right on the Signal's name......

......then choose Your desired colour!

3.5. Changing the Simulation Time

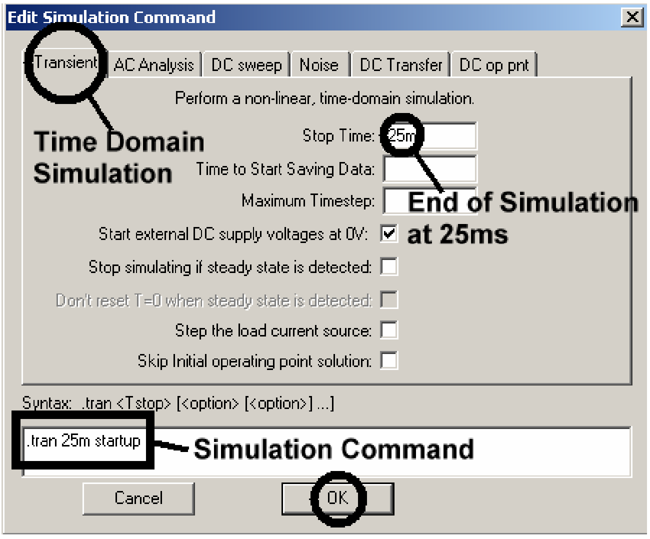


.tran 25m startup

This simple line ist the simulation command! It means::

Simulate all signal curves from 0 to 25ms.

(The simulation will always start at zero).



Edit Simulation Command

Perform a non-linear, time-domain simulation.

Stop Time: 25m

Time to Start Saving Data:

Maximum Timestep:

Start external DC supply voltages at 0V: ☑

Stop simulating if steady state is detected: □

Don't reset T=0 when steady state is detected: □

Step the load current source: □

Skip Initial operating point solution: □

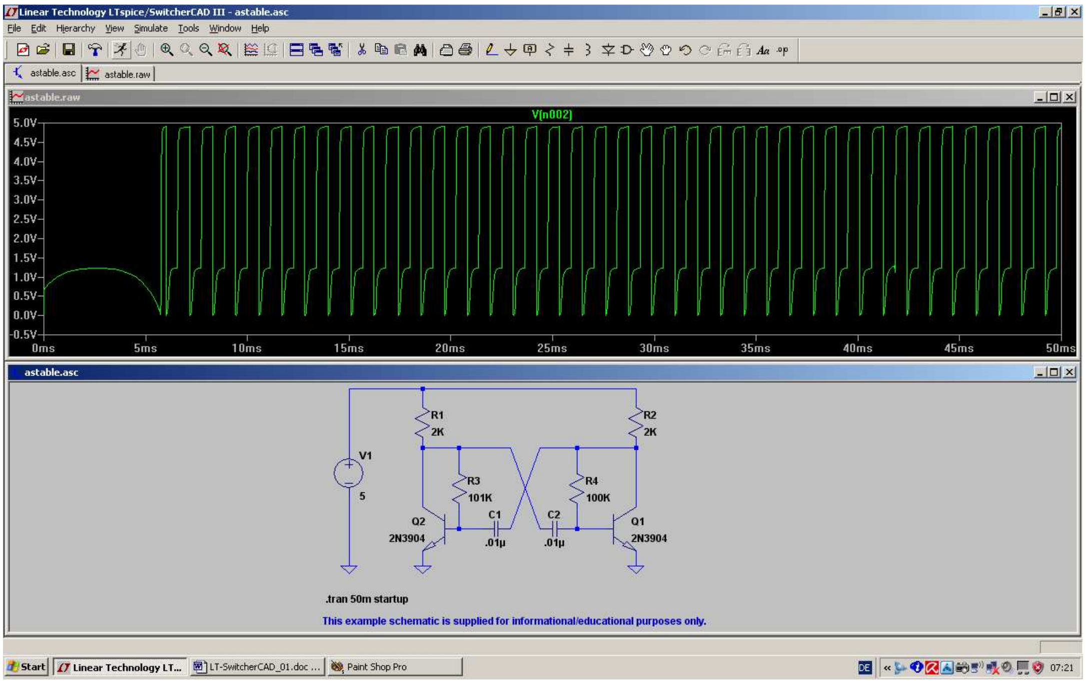
Syntax: .tran <Tstop> [<option> [<option>] ...]

.tran 25m startup

If you now move the cursor to the text of this line and right click, then you'll get this menu.

Please enter a new value for "Stop Time" if you want a longer or shorter simulation time, press OK and finally the simulation button (with "the running guy" on it) to start PSPICE.

Task:
Please choose a stop time of 50ms and simulate the collector voltage V(n002) at the left transistor



Linear Technology LTspice/SwitcherCAD III - astable.asc

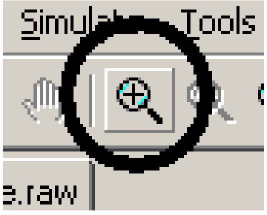
.tran 50m startup

This example schematic is supplied for informational/educational purposes only.

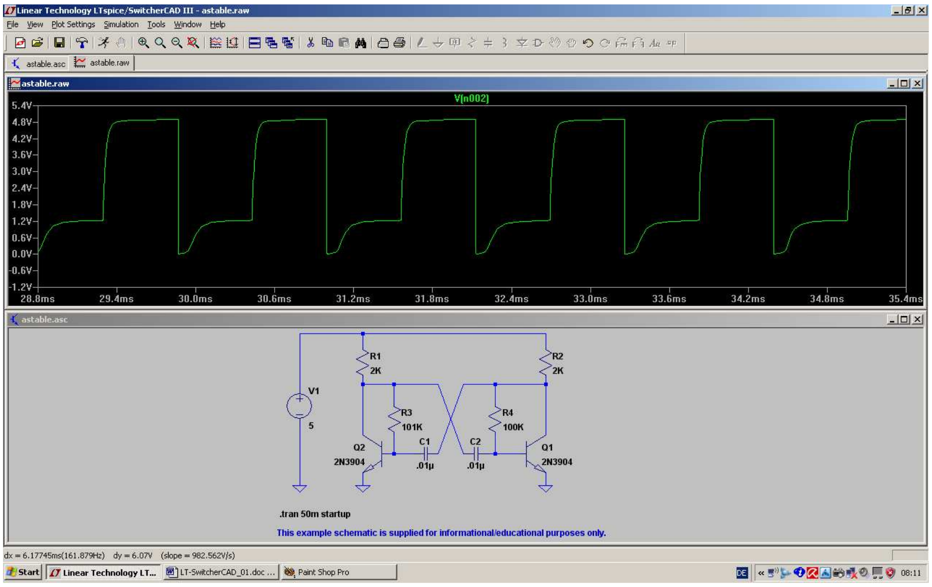
This should be the result.
You can see that the simulation of the circuit behaviour after the start is a very important point -- e. g. for the analysis of switching power supplies (...and the parts, delivered by Linear Technologies...). But if you are only interested in the „steady state", then you should use the zoom function.

Task:
Make the voltage curve visible between 30 and 35ms.

Solution:



a) Press this „enlarge button” and afterwards drag the region between 30 and 35ms with the pressed left mouse button.

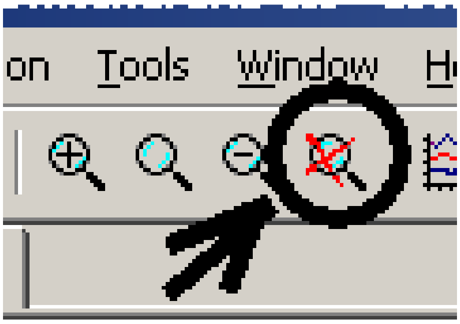


.tran 50m startup

This example schematic is supplied for informational/educational purposes only.

dx = 6.17745ms(161.879Hz) dy = 6.07V (slope = 982.562V/s)

b) When releasing the mouse button you should see this diagram.



c) If you want to go back to the full screen presentation press this button.

3.6. Changing Current or Voltage Ranges

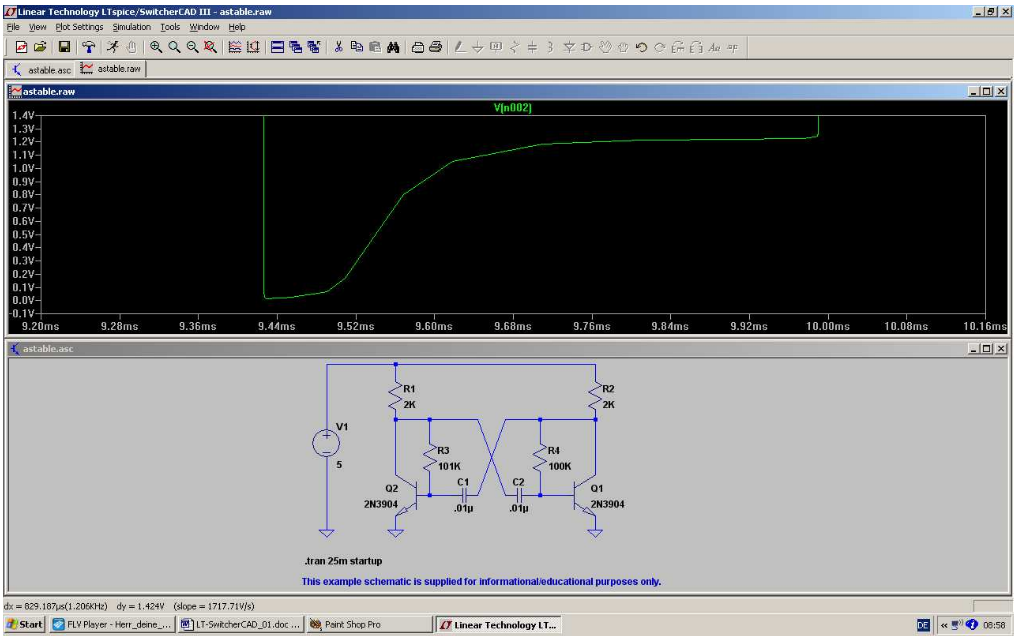
Very often you want to examine little details of a curve. So you have 2 possibilities:

a) Use the discussed zoom function from chapter 3.4.

b) Switch from “Autorange“ to „Manual Limits“.

Example for solution a):

Task:
Show only the short negative „peak“ of the collector voltage by zooming:



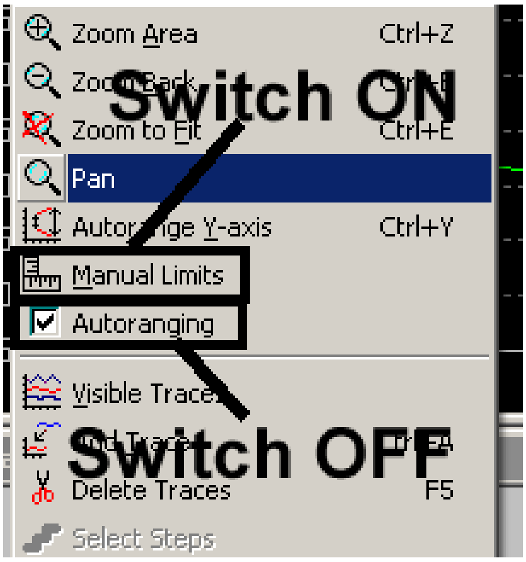
.tran 25m startup

This example schematic is supplied for informational/educational purposes only.

dx = 829.187µs(1.206KHz)   dy = 1.424V   (slope = 1717.71V/s)

Example for Solution b):

Task:
Set the values fort he vertical axis to the range 0…..+1.5V.



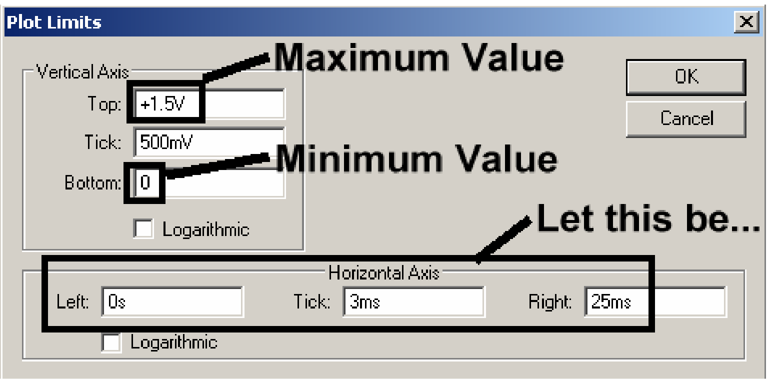
Switch ON

Switch OFF

Step 1:
RIGHT click on the calibration of the vertical axis in the diagram and you get this menu for ***Manual Setting on BOTH AXIES***.

(Attention:
If you left click on the axis scaling, you’ll only see the „quick solution menu”. Please test this, but we want to have a look on the „big solution” and continue with the case of the right mouseclick).

Now switch off “Autoranging” and activate „**Manual Limits**“ .



Plot Limits

Vertical Axis

Top: +1.5V

Tick: 500mV

Bottom: 0

□ Logarithmic

Horizontal Axis

Left: 0s

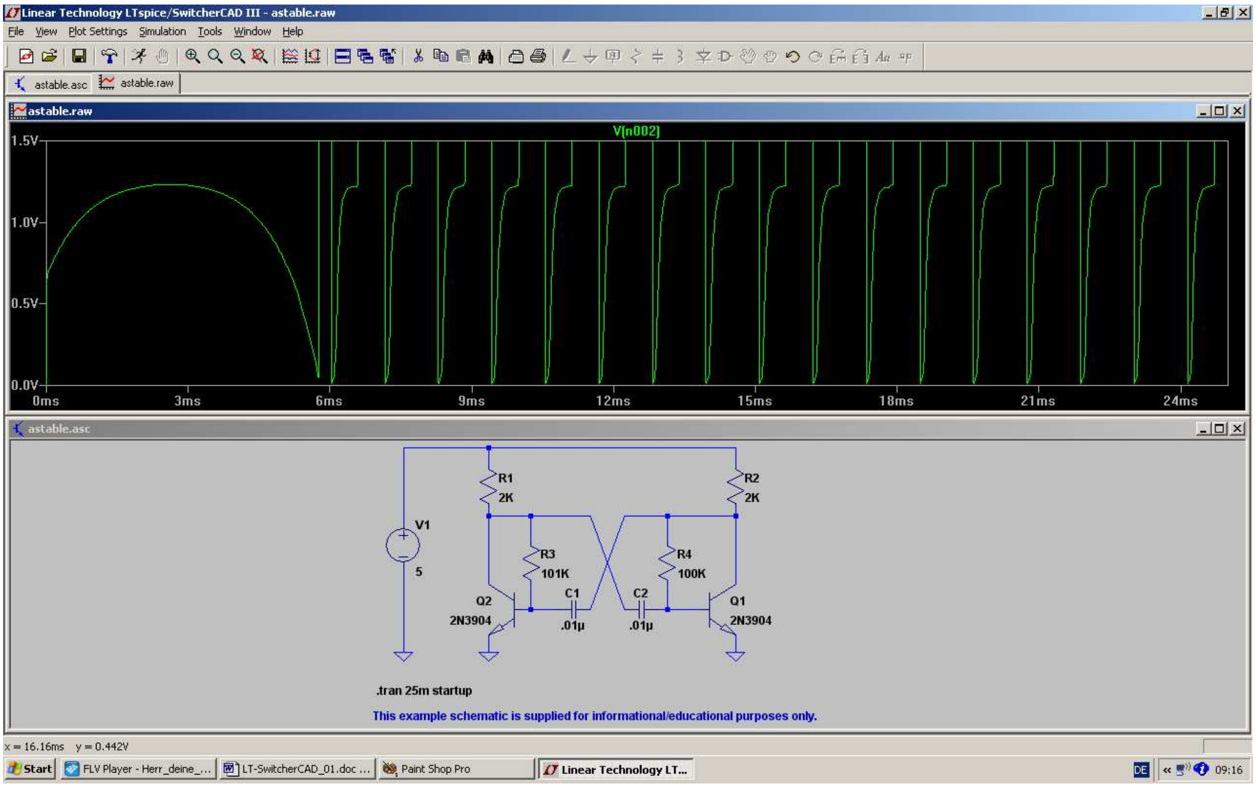
Tick: 3ms

Right: 25ms

□ Logarithmic

Step 2:
Enter the new border values for the vertical axis.

If you want you can also modify the horizontal time axis. But at this moment: let it be...



.tran 25m startup

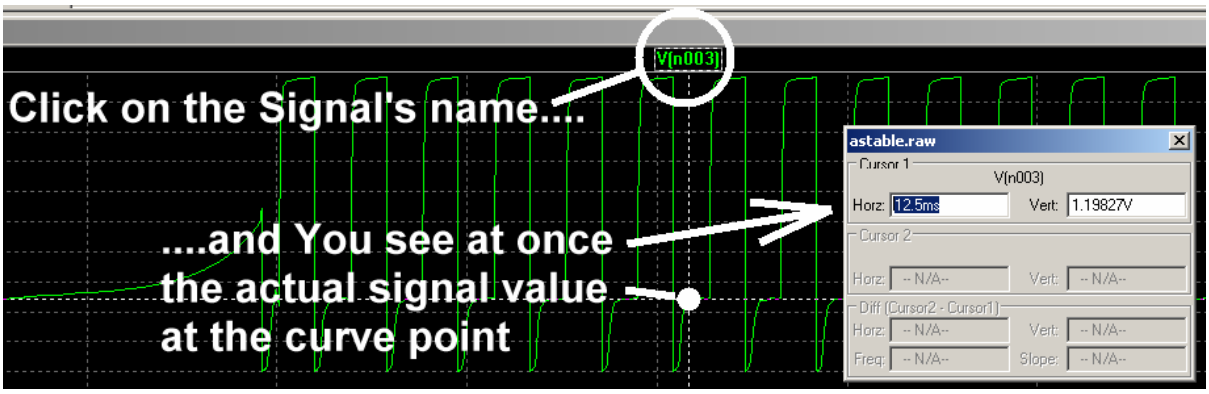
This example schematic is supplied for informational/educational purposes only.

This is the goal when returning to the full screen presentation including the schematic, where you now -- if you want! -- can modify part values.

3.7. Using Cursors

3.7.1. Working with one Cursor

Measuring the signal value at special points is easy with a cursor:



Click on the Signal's name....

....and You see at once the actual signal value at the curve point

V(n003)

Horz: 12.5ms Vert: 1.19827V

Horz: --N/A-- Vert: --N/A--

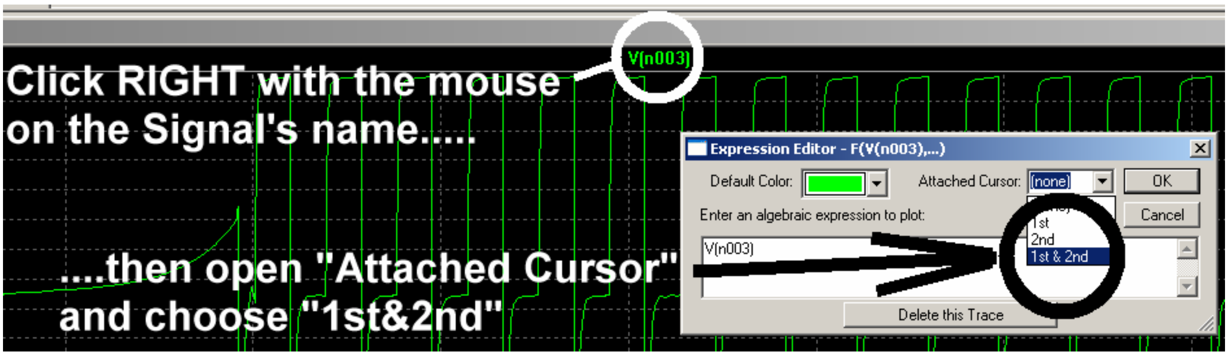
Horz: --N/A-- Vert: --N/A--

Freq: --N/A-- Slope: --N/A--

If you now move the mouse to the marked point, you suddenly see the number of the cursor (here: „1“) hanging on your mouse. By “dragging” (= pressing the left mouse button + moving the mouse) it is possible to shift the cursor along the signal’s curve. The actual curve value is then indicated in the window.

3.7.2. Working with two Cursors

RIGHT click on the signal’s name and open „Attached Cursor“ in the menu:

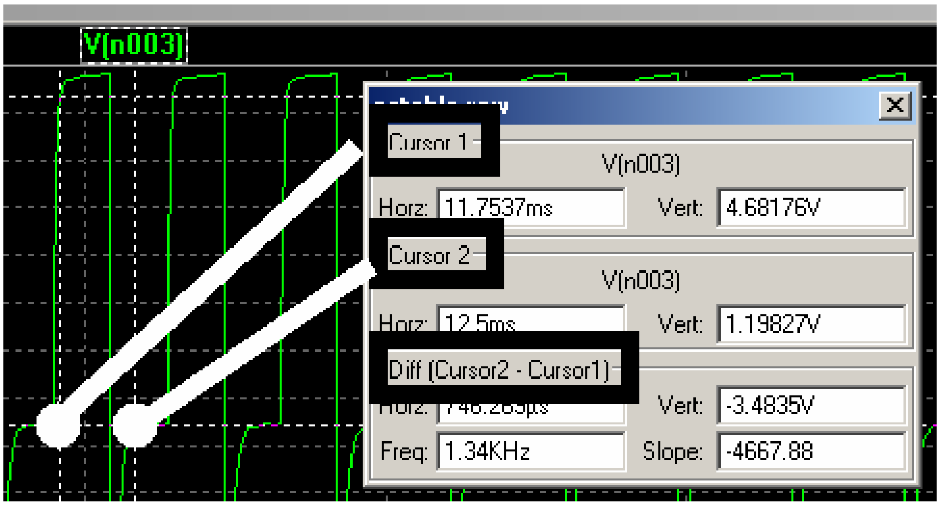


Click RIGHT with the mouse on the Signal's name.....

....then open "Attached Cursor" and choose "1st&2nd"

Enter an algebraic expression to plot:

V(n003)



V(n003)

Horz: 11.7537ms Vert: 4.68176V

V(n003)

Horz: 12.5ms Vert: 1.19827V

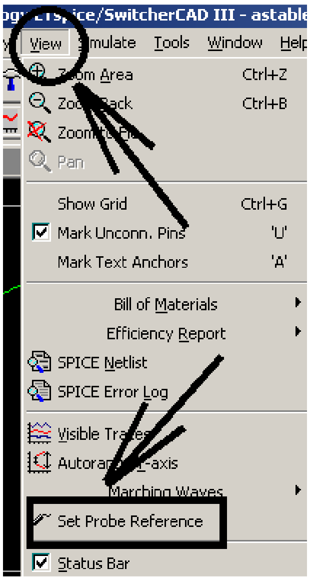
Horz: 746.265µs Vert: -3.4835V

Freq: 1.34KHz Slope: -4667.88

Now two cursors are visible and can be moved independently. The actual cursor values **AND** the differences are displayed in the window.

3.7. Differential Measurements

Every measured and presented voltage ist referred to a REFERENCE POINT (normally: ground). But how can we get the voltage between two points within the schematic? In this case you need a „differential voltmeter“, but where can you find that?

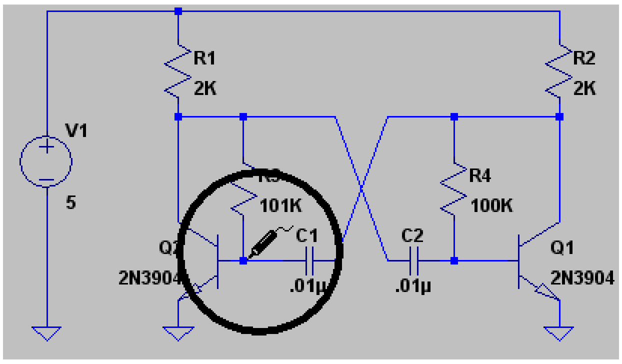


Sorry:
LTspice does not use a differential voltmeter like other SPICE programs!

You must change the REFERENCE POINT for this purpose. That means that you at first must set a new reference point at one node and then click onto another node!

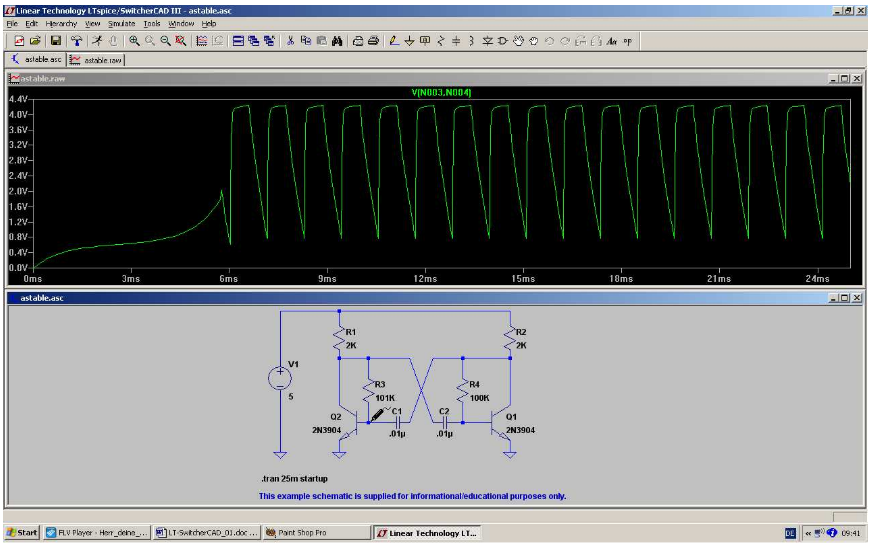
So please open the menu „**View**“ and activate „**Set Probe Reference**”.

Now we want to see the capacitor C1’s voltage (= base capacitor of the left transistor).



After OK you now have a “testclip” as cursor. Click with it on the base of the left transistor.

Then click on the collector of the right transistor (= other side of capacitor C1). That is all!



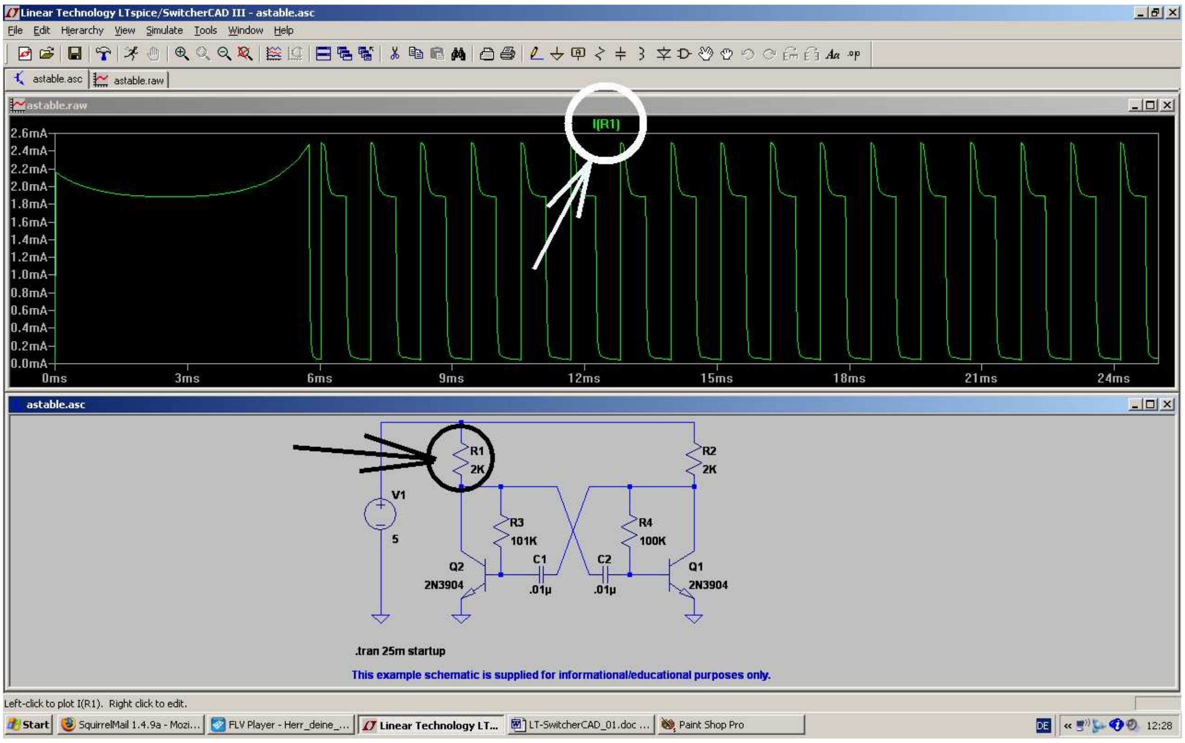
Now use -- if you want! -- the zoom function for interesting details.

To leave this “differential measurement option” simply click on ESCAPE.

3.8. Current Measurements

Make the circuit diagram window active (...by a left mouseclick on it....) and move the cursor exactly ***over the part*** of which you want to know the current. The cursor changes to a “current sensor” and with a left mouseclick on the part you see the current curve.

Example:
Current in resistor R1 (= Collector resistor of the left transistor):



```
.tran 25m startup
```

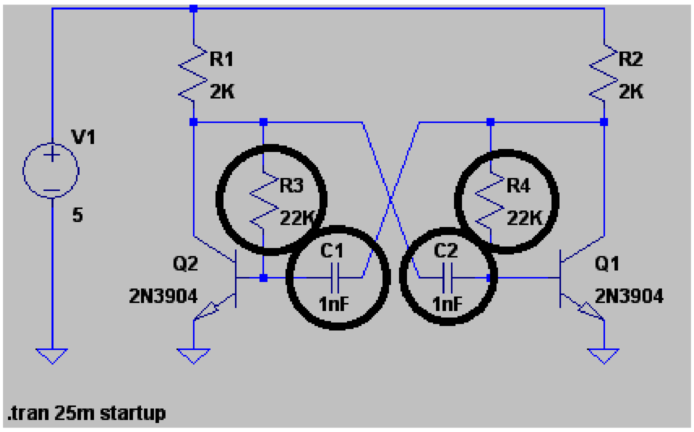
This example schematic is supplied for informational/educational purposes only.

Left-click to plot I(R1). Right click to edit.

3.9. Changing Part Values

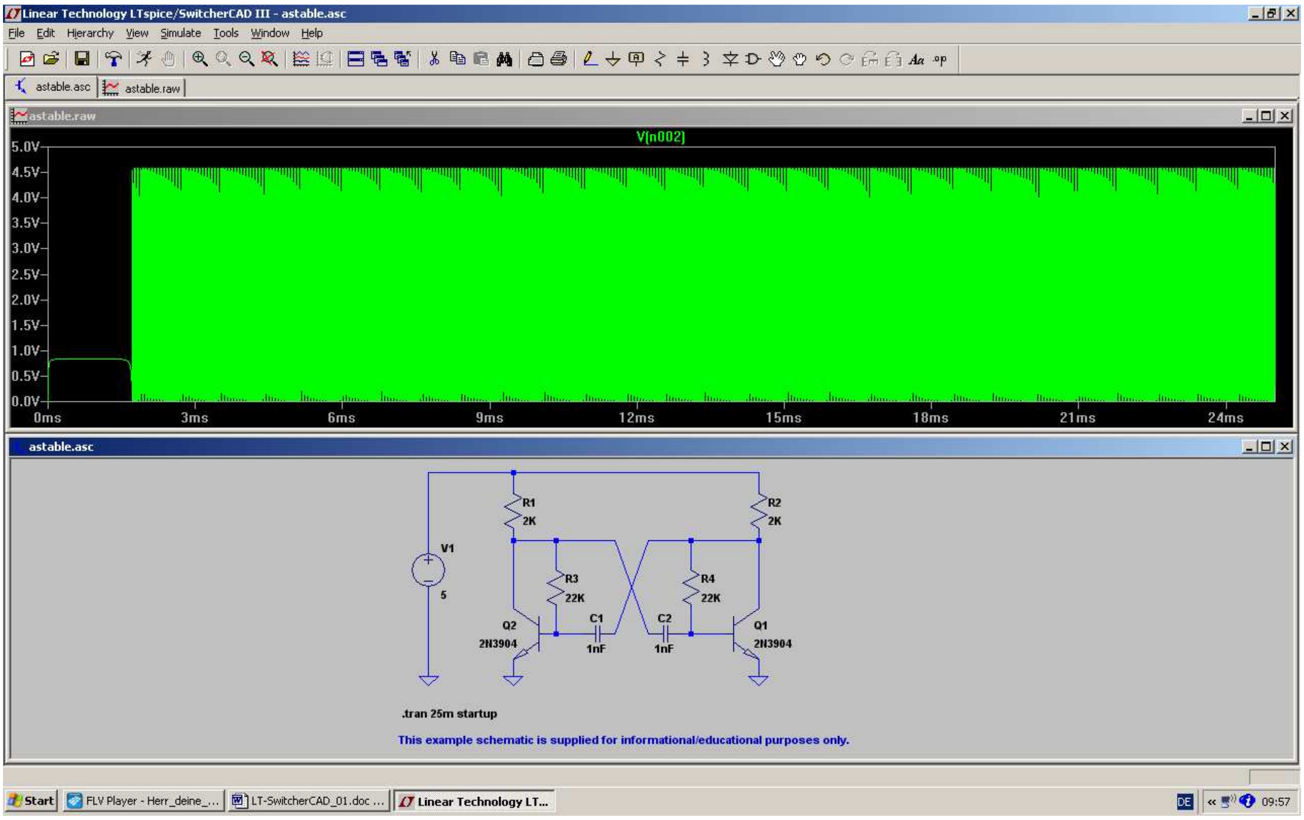
Now we want to repeat the simulation with different part values. Let every capacitor be C = 1000pF and the both base resistors of the transistors R3 = R4 = 22kΩ.

Move at every part the cursor exactly on its **value indication** and **right click with the mouse**. So you get the property menu and you can modify the value. Close with OK.



This is the new circuit to be simulated….

….and this should be the simulation result:



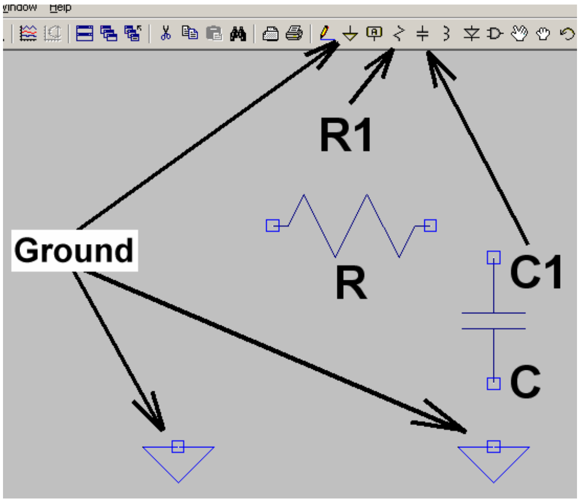
Please use the zoom function or modify the simulation time to see the details or to measure the new oscillation frequency.

----

4. Project 1: RC Lowpass Filter

4.1. Drawing the Circuit Diagram with the Editor

Open a new file and save it in a new folder (e. g. named „RC-LPF_01“).



Then press the „resistor button“ to get the resistor, turn the symbol left with

<CTRL> + <R>

for 90 degrees an place it on your sheet. Leave this function either by a right mouseclick or by pressing „ESCAPE“ on the keyboard.

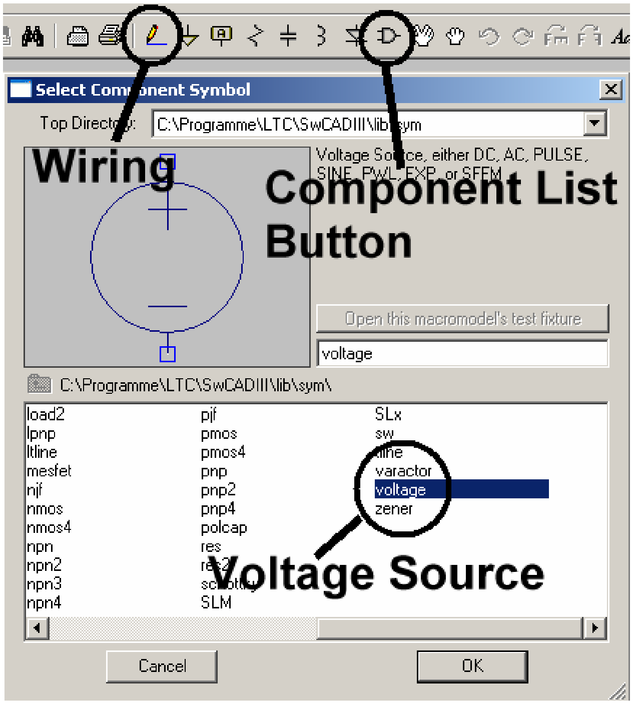
Repeat this procedure for the capacitor and two ground symbols.

Two tips:

a) To delete a part or a connection in the circuit, press **F5**. At once you get scissors as cursor. Move these scissors over the symbol (or the wiring connection) and left click: the part or the connection disappears at once.

b) To move a part, press **F7** and the cursor changes to a hand. Move this hand over the symbol of the part and left click . Now the part is connected to the cursor and can be moved with the mouse. A new left mouseclick will fix it in the new position.

Do never forget:
Leaving these special functions is done by pressing the ESCAPE KEY on the keyboard!



Top Directory: C:\Programme\LTC\SwCADIII\lib\sym

Voltage Source, either DC, AC, PULSE, SINE, PWL, EXP, or SFFM

Open this macromodel's test fixture

voltage

C:\Programme\LTC\SwCADIII\lib\sym\

load2, lpnp, ltline, mesfet, njf, nmos, nmos4, npn, npn2, npn3, npn4, pjf, pmos, pmos4, pnp, pnp2, pnp4, polcap, res, res2, schottky, SLM, SLx, sw, tline, varactor, voltage, zener

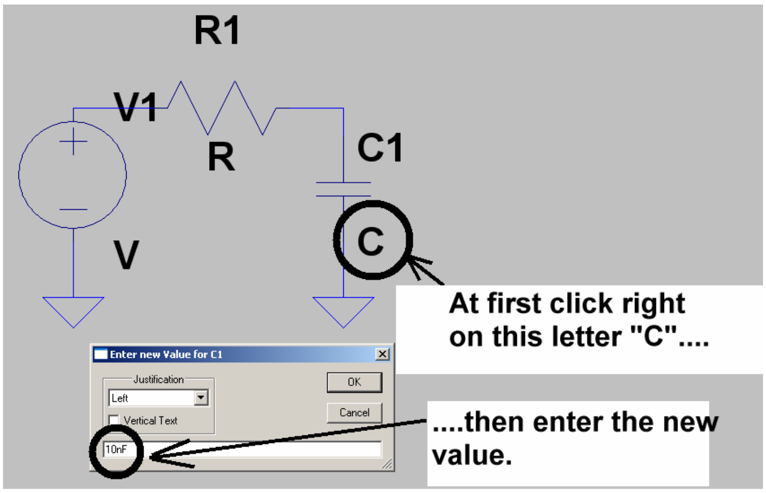
We still need a voltage source and you must therefore open the „**component list**“ by pressing the special button.

In the appearing list please look for „**voltage**“, mark it with a mouseclick and press OK. The symbol of the voltage source is then hanging on the cursor and can be placed within the schematic.

Now we can start with wiring. Press the button with the pencil, roll the cursor to the start of the connection, click left and roll him to the other point. One more left mouseclick and the connection is perfect!

Important:
For an „edged“ wiring start as before, roll to the edging point and left click. Then you can do the rest of wiring in another direction.

4.2. Changing Part Values



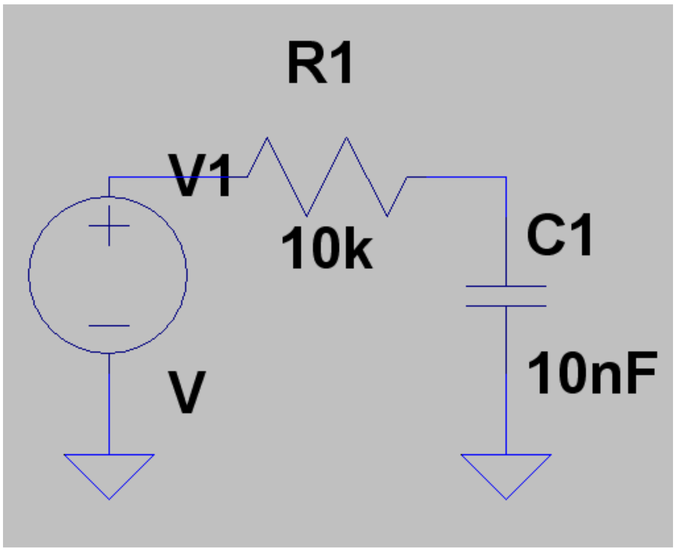
At first click right on this letter "C"....

....then enter the new value.

This is demonstrated by capacitor C1:

Move the cursor on the letter “C” and right click. Then enter the new value and close with OK.

Please change the value of the resistor R1 to 10k in the same manner.



So the circuit should now look like.

But: before programming the voltage source we must decide wether we want a

time domain

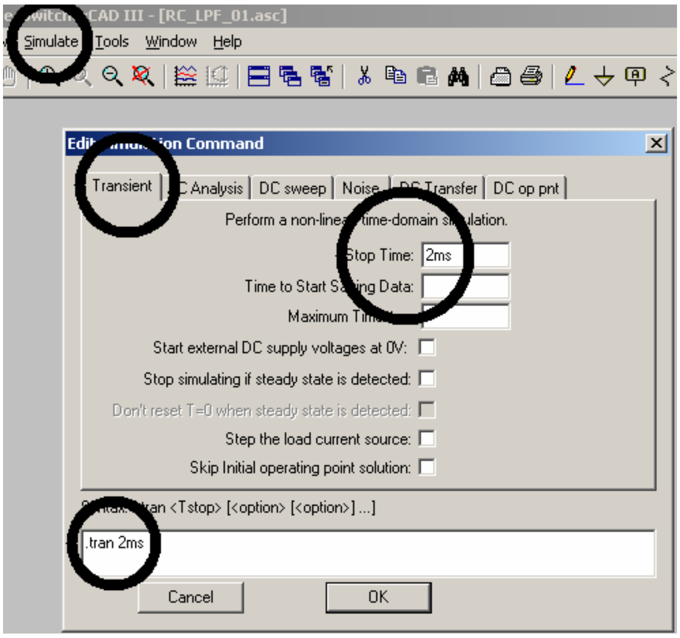
or a

frequency domain-simulation!

4.3. Time Domain Simulation: Non Repetitive Signals at the Input

4.3.1. Step Response

Please use a **transient simulation in the time domain and let the input voltage jump in a very short time from 0 to +1V.**



Perform a non-linear, time-domain simulation.

Stop Time: 2ms

Time to Start Saving Data:

Maximum Timestep:

Start external DC supply voltages at 0V: □

Stop simulating if steady state is detected: □

Don't reset T=0 when steady state is detected: □

Step the load current source: □

Skip Initial operating point solution: □

Syntax: .tran <Tstop> [<option> [<option>] ...]

.tran 2ms

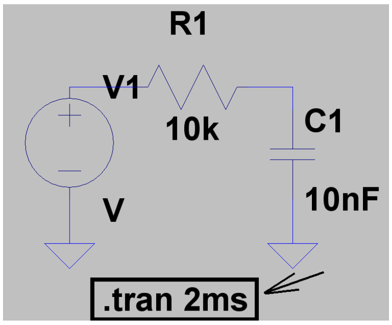
Click on the button „**Simulate**“, then „**Edit Simulation Command**“ and „**Transient**“

Enter a **stop time of 2ms** and check that this was accepted by filling the lower empty field with

.tran 2ms

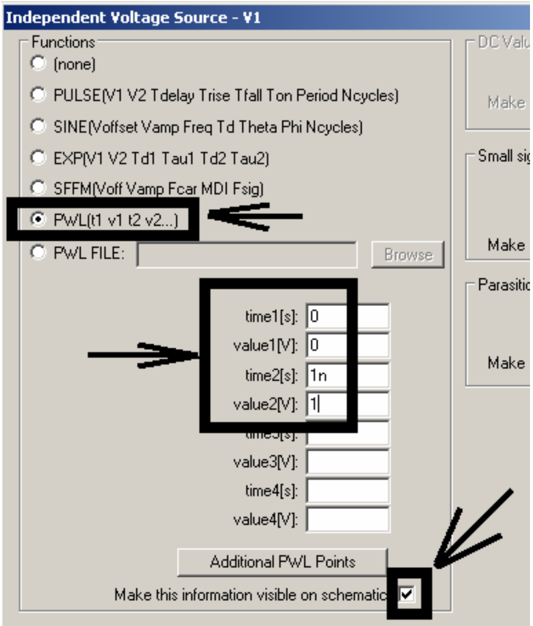
Then press OK.

.



This command line hangs now on the cursor and must be placed by a mouse click somewhere in the Editor screen.

To get the correct input voltage, right click on the symbol of the voltage aource to open the property window. There open the **„ADVANCED“ – menu.**



○ (none)

○ PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)

○ SINE(Voffset Vamp Freq Td Theta Phi Ncycles)

○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)

○ SFFM(Voff Vamp Fcar MDI Fsig)

● PWL(t1 v1 t2 v2...)

○ PWL FILE:

time1[s]: 0

value1[V]: 0

time2[s]: 1n

value2[V]: 1

time3[s]:

value3[V]:

time4[s]:

value4[V]:

Make this information visible on schematic: ☑

The input voltage curve is generated by a series of linear lines (= „pieces“) between fixed points. This is done by the

PWL = piecewise linear

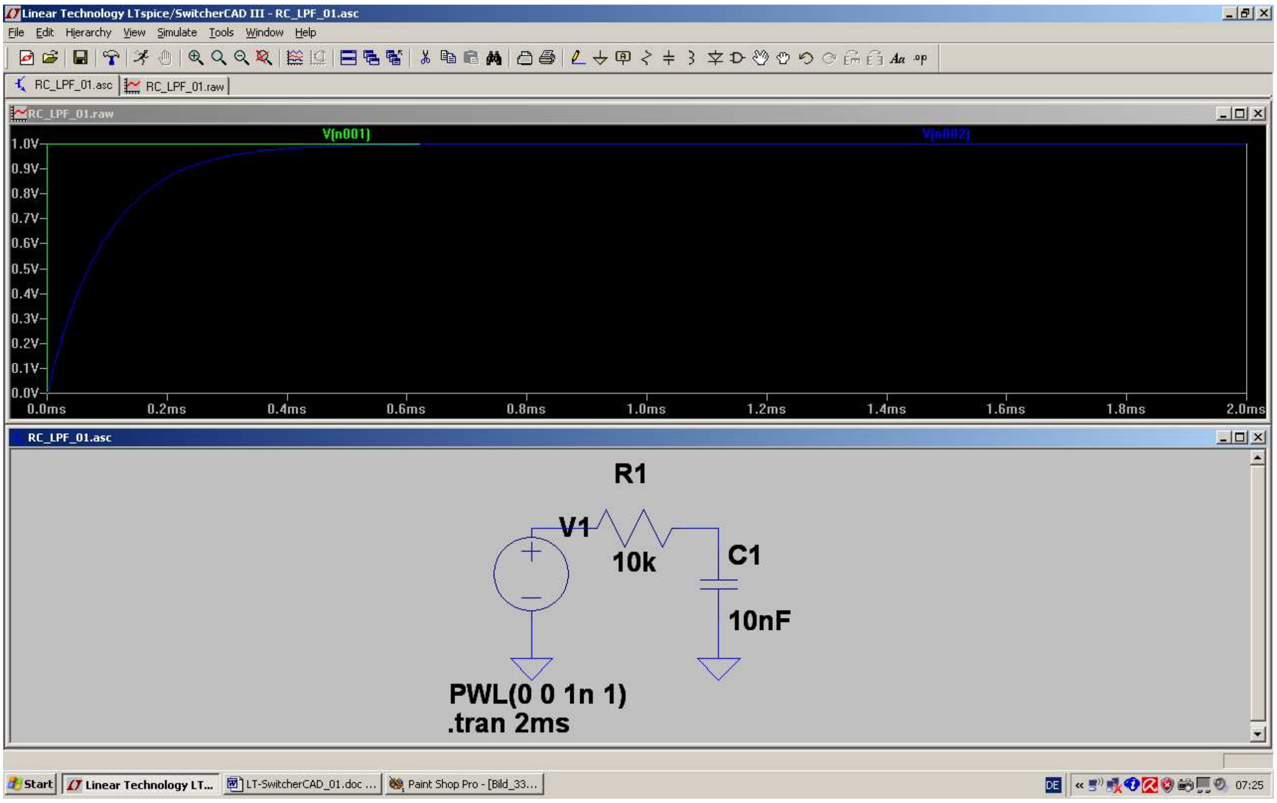
option, which now must be activated.

At time1 = 0	**the voltage is also 0V.**
At 1 ns	**the voltage value has risen to +1V and this value must not change.**

Please enter these two pairs of values in the list of the menu, make the information visible on the screen and close with OK.

Place this information on the screen. Move it -- if necessary! -- by pressing F7, clicking on it and dragging with the mouse. Leave the F7 function by pressing the ESCAPE key on the keyboard.

Now start the simulation, click on the input resp. output point in the schematic and admire the result:

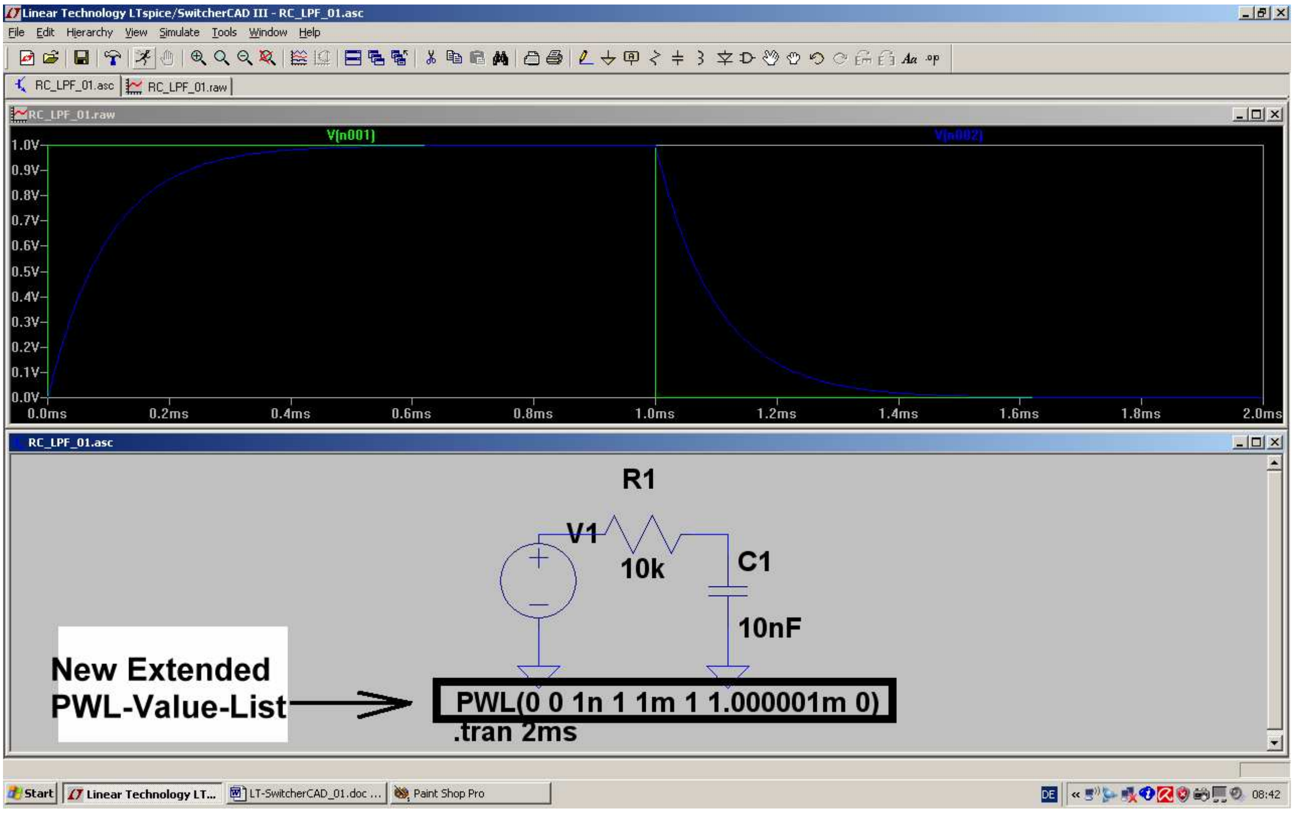


PWL(0 0 1n 1)
.tran 2ms

4.3.2. Switching ON and OFF

No problem because you have only to extend the value-pair list of the PWL source:

0	**0**	(Zero Volt at start point)
1n	**1**	(+1 Volt after 1 nanosecond)
1m	**1**	(still +1 Volt during 1 millisecond)
1.000001m	**0**	(Within 1 nanosecond back to 0V)

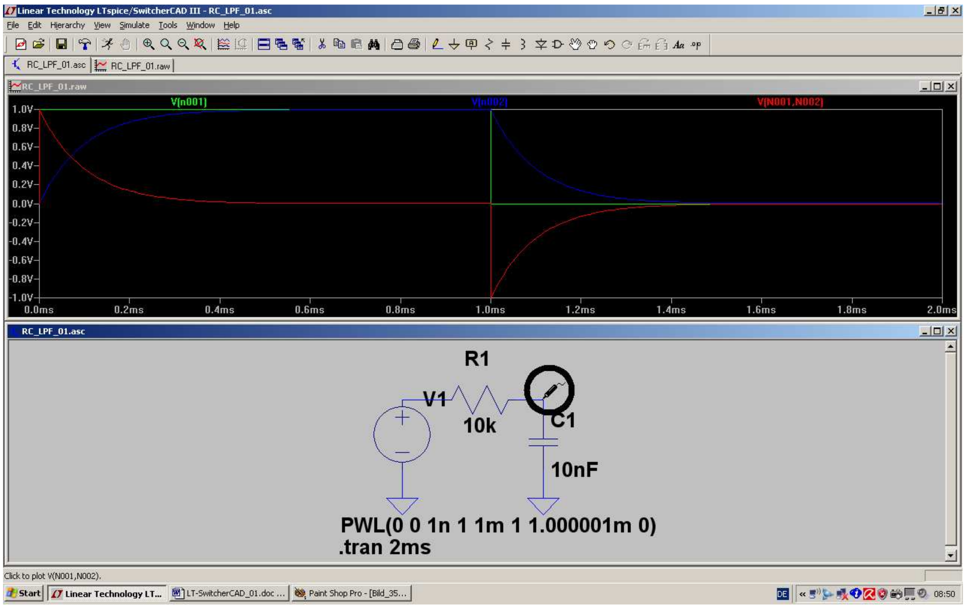


New Extended PWL-Value-List

PWL(0 0 1n 1 1m 1 1.000001m 0)
.tran 2ms

Task:
Add now the voltage at the resistor to this diagram!

Solution: add a reference point to the output (= connection between the resistor and capacitor). Touch the connection between voltage source and resistor with the cursor and click. At once the desired resistor voltage is presented in the diagram.



PWL(0 0 1n 1 1m 1 1.000001m 0)
.tran 2ms

Click to plot V(N001,N002).

4.3.3. The Impulse Response

Fourier, Fast Fourier and Laplace Transformations are important tools for modern Communication or Control Techniques. With these tools you can get all informations about the system's reaction to any input voltage form -- if you know the ***Transfer Function***.

This transfer function is the ratio of the output sine voltage to an applied input sine voltage, but presented in a complex "magnitude + phase" – form.

So let us have a look at the **impulse response**:

If you apply a „Dirac - Impulse" to the input of an unknown system, then the answer (= Impulse Response) at the output will contain every information about this system -- because this answer is a "time domain form" of the transfer function.

And with the transfer function you can -- by "convolution" -- calculate the output signal for every possible input signal.

The Dirac Impulse is a little crazy: the amplitude is infinite, the pulse length is infinitely short...but the "pulse area" (= magnitude x pulse length) is "1".

This is impossible to generate but you can use a definite limited amplitude and a definite short pulse length -- only be shure that you have again a pulse area of "1" and the pulse length is much shorter than the system's time constant (....a factor of 100 to 1000 will do the job). And in real linear systems it is allowed to reduce the pulse magnitude to a value which is not dangerous for the system itself. Afterwards simply find out the necessary correction coefficient to get again the "ideal" values.

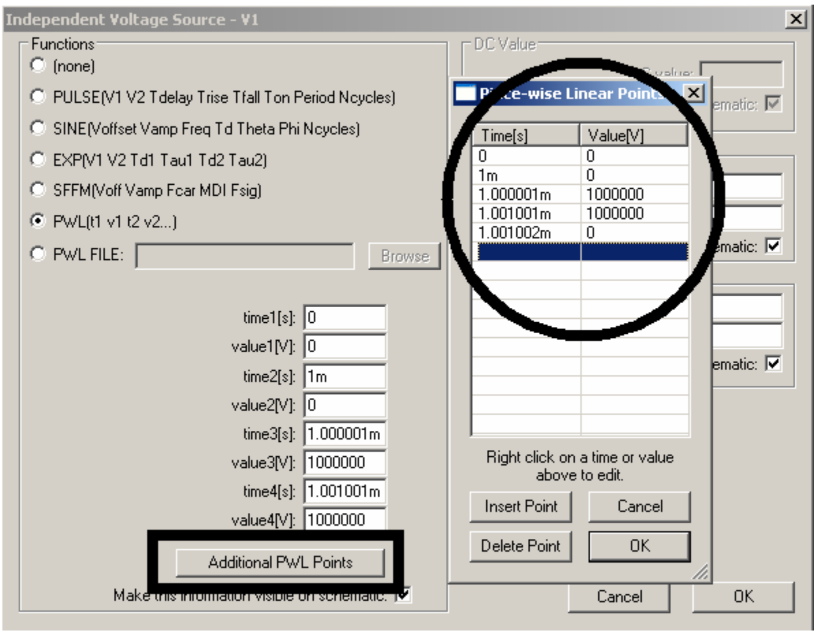
Let us have a look at our lowpass filter.

Task 1): Apply at the input of the LPF (R = 10kΩ, C = 10nF) a Dirac impulse with a peak value of Vampl = 1 Megavolt. Start with a delay time of 1 ms at an amplitude of 0V. Then the pulse rises to 1 Megavolt within 1 Nanosecond, stays on this value for 1 Microsecond and decreases to 0V within 1 Nanosecond.
So we get a pulse area of

1 Megavolt x 1Microsecond = 1Voltsecond

Simulate the impulse response for a stop time of 2ms

Task 2): Reduce the pulse amplitude to 100 Kilovolt but use a pulse length of 10 Microseconds to get again a pulse area of 1 Voltsecond. Simulate and compare the result with task 1.



Functions
○ (none)
○ PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)
○ SINE(Voffset Vamp Freq Td Theta Phi Ncycles)
○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)
○ SFFM(Voff Vamp Fcar MDI Fsig)
● PWL(t1 v1 t2 v2...)
○ PWL FILE:

time1[s]: 0
value1[V]: 0
time2[s]: 1m
value2[V]: 0
time3[s]: 1.000001m
value3[V]: 1000000
time4[s]: 1.001001m
value4[V]: 1000000

Make this information visible on schematic: ☑

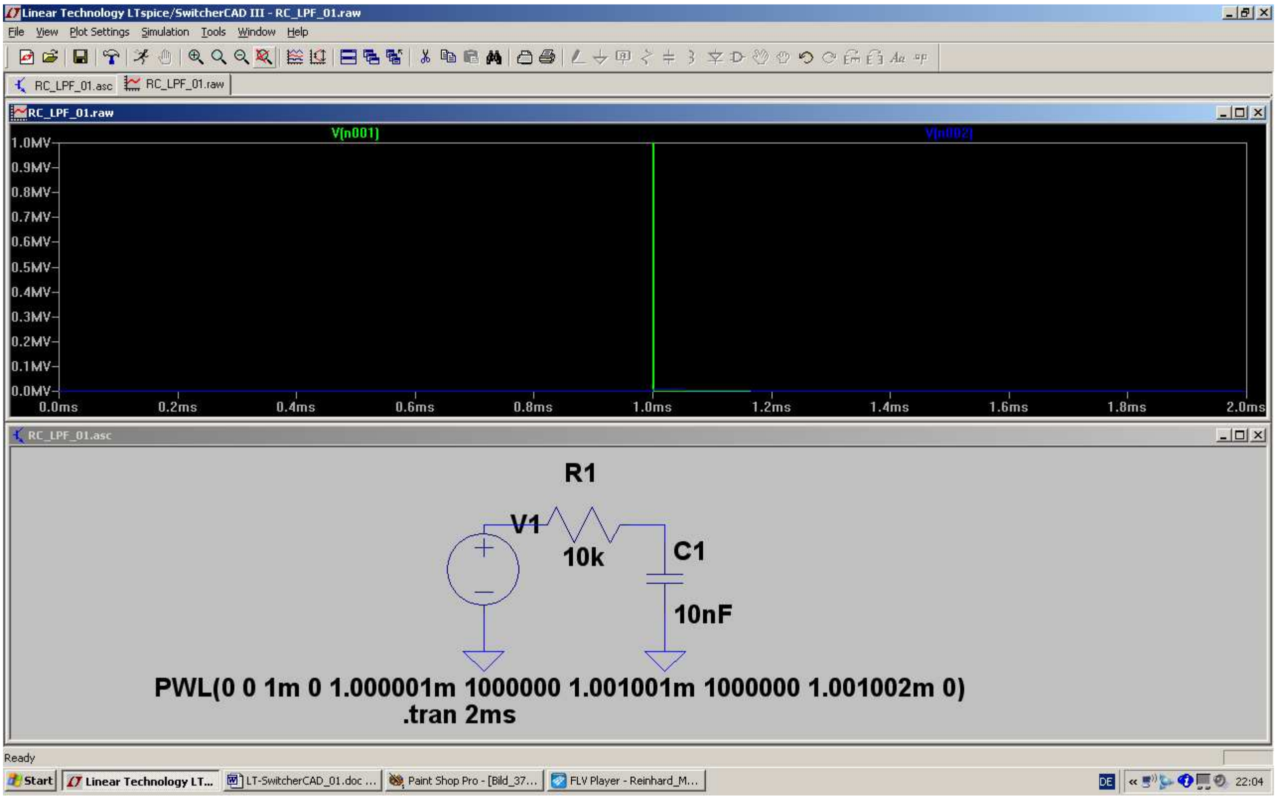
Piece-wise Linear Points

Time[s]	Value[V]
0	0
1m	0
1.000001m	1000000
1.001001m	1000000
1.001002m	0

Solution for Task 1:
First right click on the symbol of the voltage source to open the property menu. Then fill the PWL-list with the informations about the additional points and value pairs (use the button “Additional PWL Points”):

0	**0**
1m	**0**
1,000001m	**1000000**
1,001001m	**1000000**
1,001002m	**0**

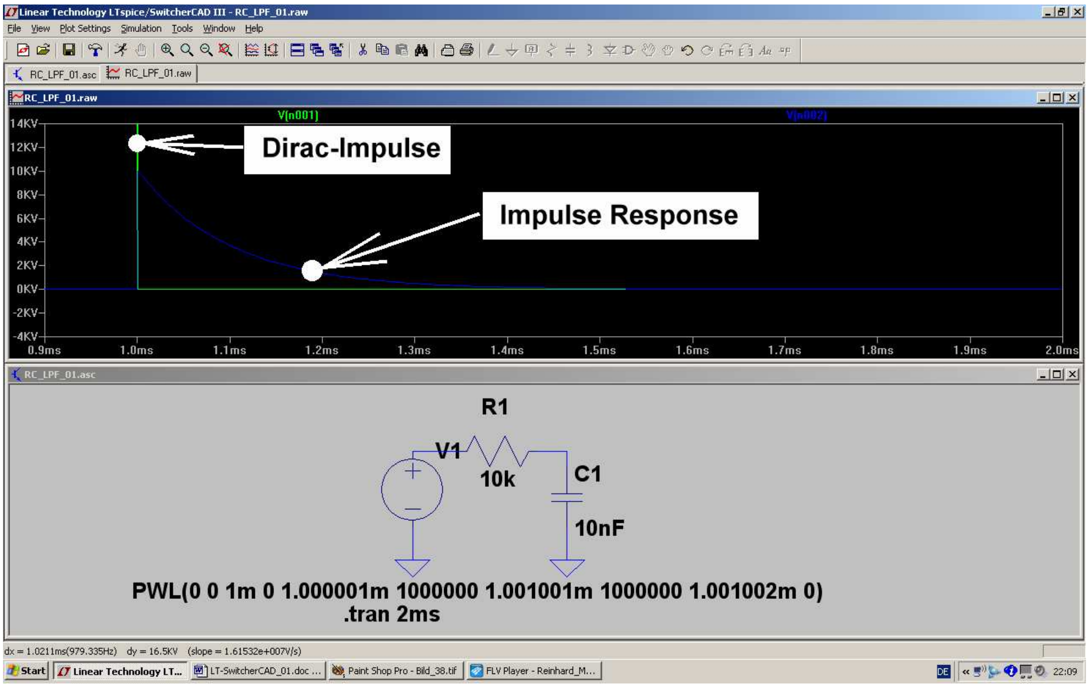
When simulating input and output voltage you get this screen where only the Dirac impulse with its amplitude of 1 Megavolt can be seen:



PWL(0 0 1m 0 1.000001m 1000000 1.001001m 1000000 1.001002m 0)
.tran 2ms

Use the zoom function for other details, because the system’s response has only an amplitude of 10KV. So change the scaling of the two axis to see the response curve -- and afterward have a look at the solution on the following page.

This is the correct presentation:

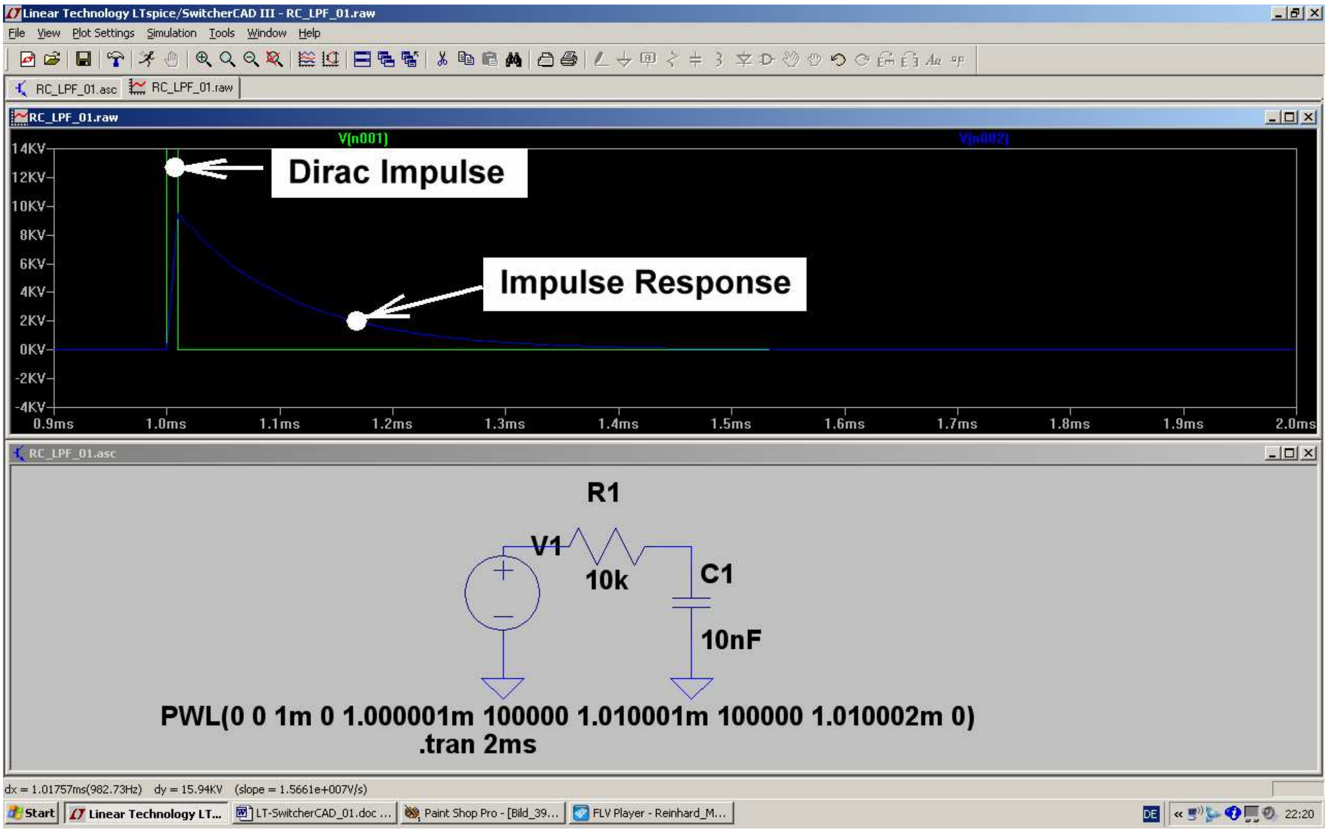


```
PWL(0 0 1m 0 1.000001m 1000000 1.001001m 1000000 1.001002m 0)
.tran 2ms
```

```
dx = 1.0211ms(979.335Hz)    dy = 16.5KV    (slope = 1.61532e+007V/s)
```

Solution for Task 2:

Please be aware of the new pulse length (10 microseconds) and amplitude (100KV) and the modified PWL list in the property menu of the voltage source!



```
PWL(0 0 1m 0 1.000001m 100000 1.010001m 100000 1.010002m 0)
.tran 2ms
```

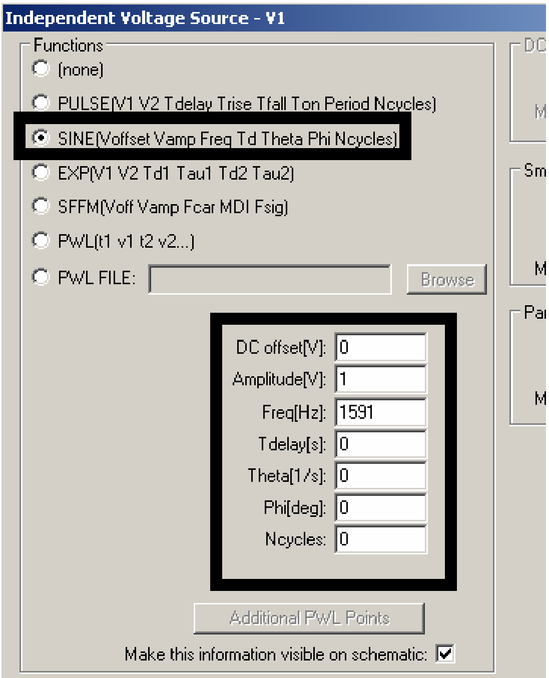
```
dx = 1.01757ms(982.73Hz)    dy = 15.94KV    (slope = 1.5661e+007V/s)
```

4.4. Time Domain Simulation using Periodic Signals at the Input

4.4.1. Sine Wave (f = 1591 Hz)

We use a sine wave with a peak value of 1V at the corner frequency of this LPF. The corner frequency can be calculated in this manner:

$$f_{Corner} = \frac{1}{2 \bullet \pi \bullet R \bullet C} = \frac{1}{2 \bullet \pi \bullet 0.1ms} = 1591Hz$$



Independent Voltage Source - V1

Functions

○ (none)

○ PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)

● SINE(Voffset Vamp Freq Td Theta Phi Ncycles)

○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)

○ SFFM(Voff Vamp Fcar MDI Fsig)

○ PWL(t1 v1 t2 v2...)

○ PWL FILE:

DC offset[V]:	0
Amplitude[V]:	1
Freq[Hz]:	1591
Tdelay[s]:	0
Theta[1/s]:	0
Phi[deg]:	0
Ncycles:	0

Make this information visible on schematic: ☑

Open the property menu of the voltage source by clicking on the source symbol and choose “SINE”. Then enter the following properties:

DC Offset **= 0**

Amplitude **= 1 Volt (= peak value)**

Frequency **= 1591 Hz**

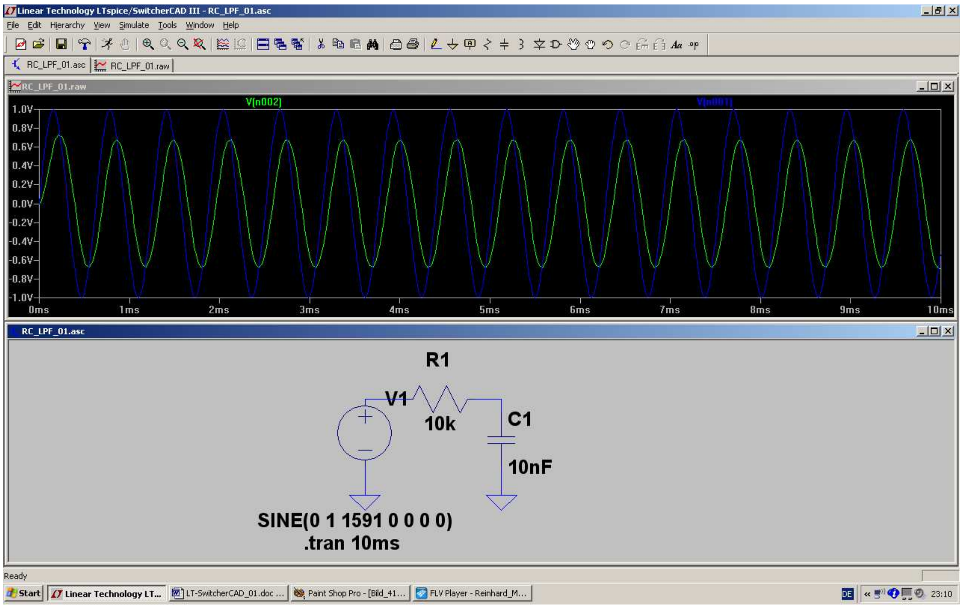
Tdelay **= 0 (= delay time after start)**

Theta **= 0 (= damping factor. Enter zero for a constant sine wave)**

Phi **= 0 (= start value of phase)**

Ncycles **= 0 or empty field (this entry is important if you want to create a „Burst Signal“)**

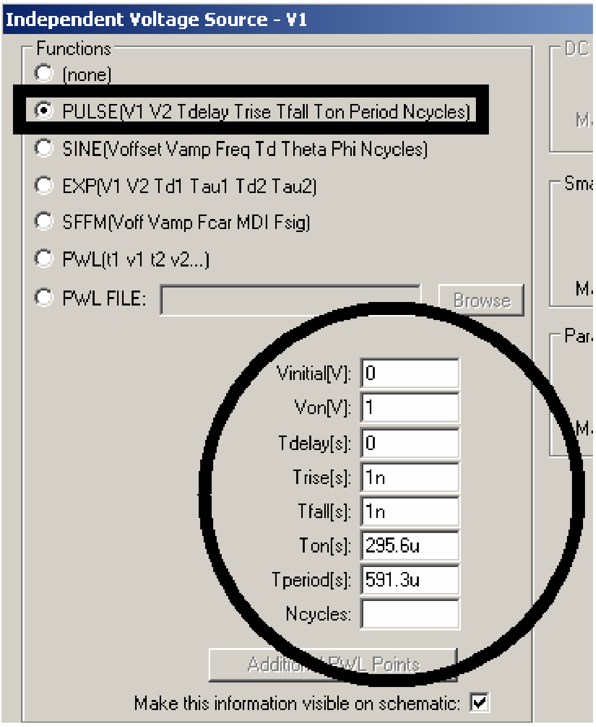
Now open the „Simulate“ menu and „**Edit Simulation Command**“. Enter a stop time of 10 ms and then start the simulation. Check the values of 45 degrees of phase shift and an amplitude degration to 70% at the corner frequency on the screen.



Linear Technology LTspice/SwitcherCAD III - RC_LPF_01.asc

SINE(0 1 1591 0 0 0 0)
.tran 10ms

4.4.2. Square Wave (f = 1691 Hz)



Independent Voltage Source - V1

Functions

○ (none)

● PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)

○ SINE(Voffset Vamp Freq Td Theta Phi Ncycles)

○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)

○ SFFM(Voff Vamp Fcar MDI Fsig)

○ PWL(t1 v1 t2 v2...)

○ PWL FILE: [Browse]

Vinitial[V]:	0
Von[V]:	1
Tdelay[s]:	0
Trise[s]:	1n
Tfall[s]:	1n
Ton[s]:	295.6u
Tperiod[s]:	591.3u
Ncycles:	

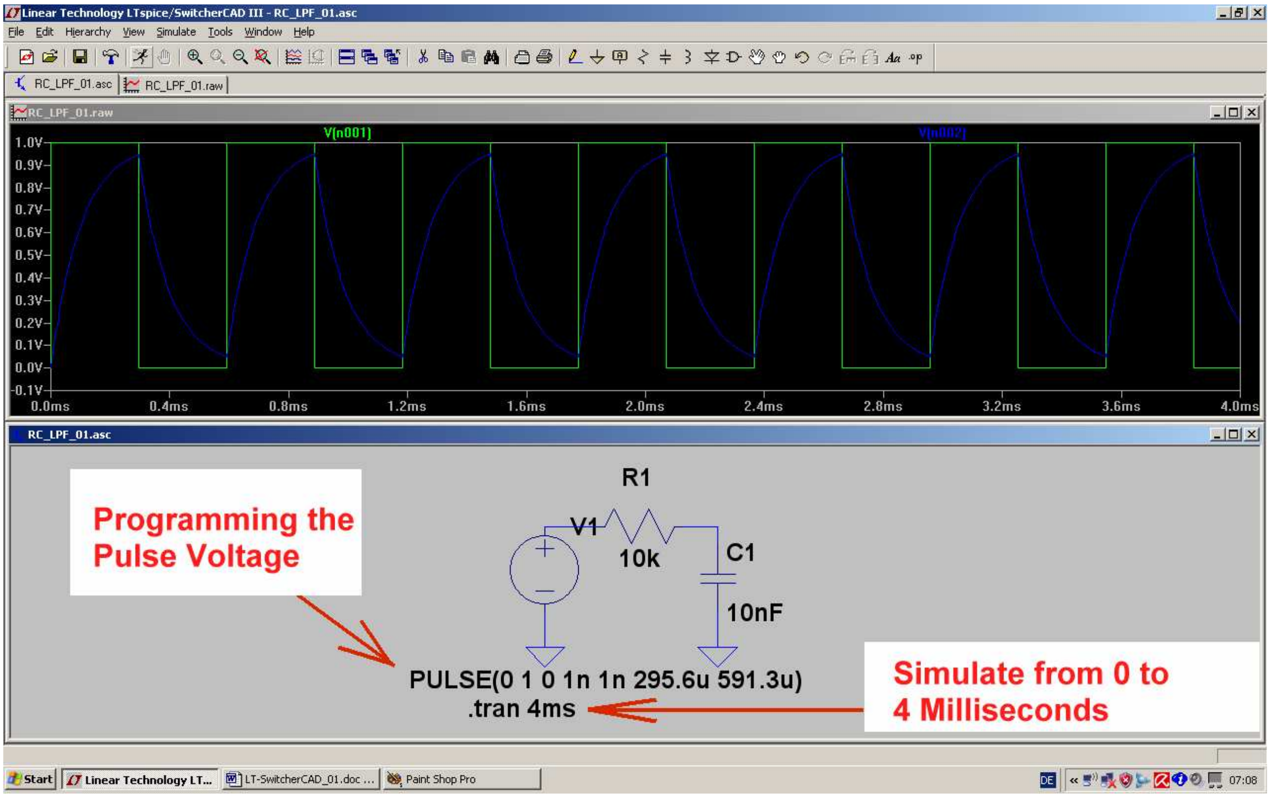
Make this information visible on schematic: ☑

Start with the properties of the voltage vource.

For „PULSE“ we need the following entries:

Vinitial	**= 0 (= minimum voltage value)**
Von	**= 1 Volt (= maximum voltage value)**
Tdelay	**= 0 (= delay after start)**
Trise	**= 1 Nanosecond (= rise time)**
Tfall	**= 1 Nanosecond (= fall time)**
Ton	**= 295,6 Microseconds (= pulse length)**
Tperiod	**= 591,3 Microseconds(= Period Time)**
Ncycles	**= 0 or empty field (= number of cycles when programming a burst signal)**

With a stop time of 4ms you get this result:



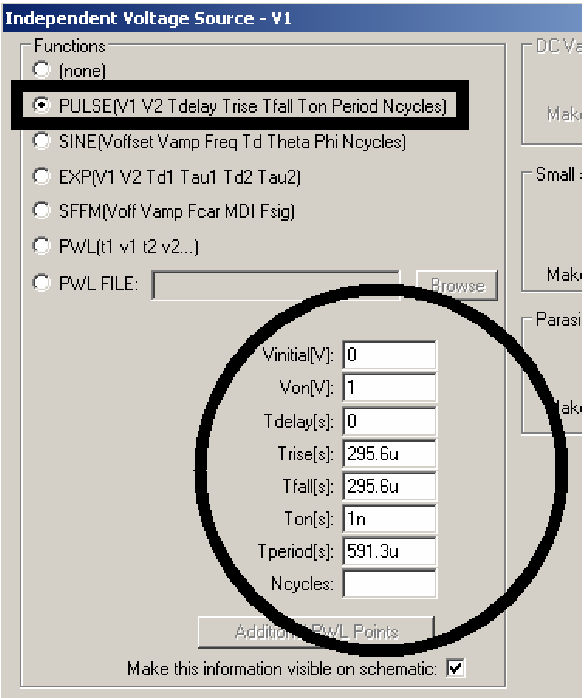
Programming the Pulse Voltage

PULSE(0 1 0 1n 1n 295.6u 591.3u)
.tran 4ms

Simulate from 0 to 4 Milliseconds

4.4.3. Triangle Wave (f = 1691 Hz)

This is a little bit difficult, because we have to „strangulate“ the Pulse Source:
If we choose a very long rise and fall time but a short pulse length then we get exactly that what we need: a triangle wave.



Independent Voltage Source - V1

Functions

○ (none)
● PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)
○ SINE(Voffset Vamp Freq Td Theta Phi Ncycles)
○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)
○ SFFM(Voff Vamp Fcar MDI Fsig)
○ PWL(t1 v1 t2 v2...)
○ PWL FILE:

Vinitial[V]: 0
Von[V]: 1
Tdelay[s]: 0
Trise[s]: 295.6u
Tfall[s]: 295.6u
Ton[s]: 1n
Tperiod[s]: 591.3u
Ncycles:

Make this information visible on schematic: ☑

So please enter the following properties:

Vinitial = 0 (= minimum voltage value)

Von = 1 Volt (= maximum voltage value)

Tdelay = 0 (= delay after start)

Trise = 295,6 Microseconds (= rise time)

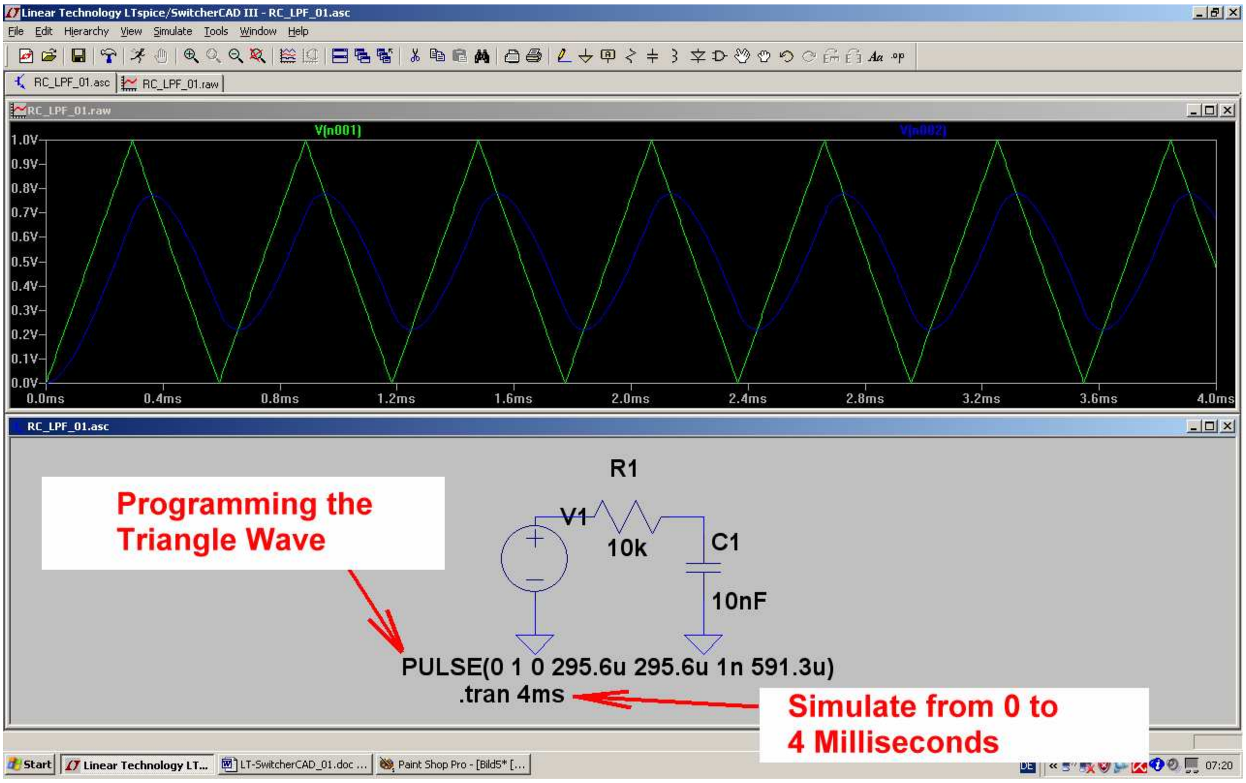
Tfall = 295,6 Microseconds (= fall time)

Ton = 1 Nanosecond (= pulse length)

Tperiod = 591,3 Microseconds(= period time)

Ncycles = 0 or empty field (= number of cycles for a burst signal)

This is the result for a stop time of 4ms:

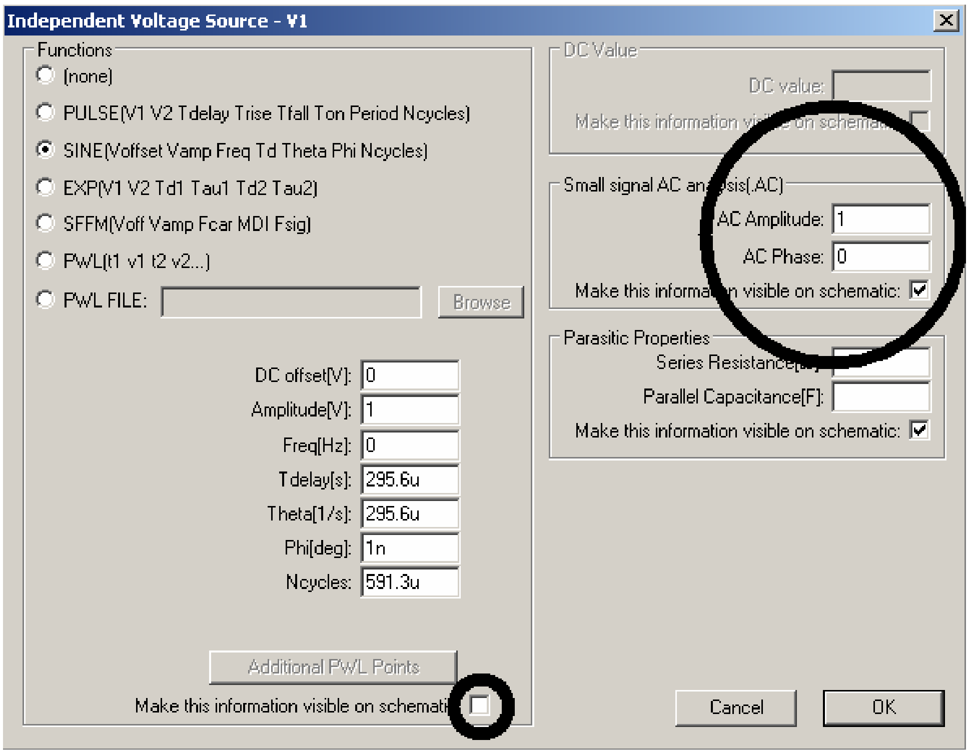


PULSE(0 1 0 295.6u 295.6u 1n 591.3u)
.tran 4ms

Programming the Triangle Wave

Simulate from 0 to 4 Milliseconds

4.5. Frequency Domain Simulation: AC Sweep



Independent Voltage Source - V1

Functions

○ (none)

○ PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)

● SINE(Voffset Vamp Freq Td Theta Phi Ncycles)

○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)

○ SFFM(Voff Vamp Fcar MDI Fsig)

○ PWL(t1 v1 t2 v2...)

○ PWL FILE:

DC offset[V]: 0

Amplitude[V]: 1

Freq[Hz]: 0

Tdelay[s]: 295.6u

Theta[1/s]: 295.6u

Phi[deg]: 1n

Ncycles: 591.3u

Make this information visible on schematic: □

DC Value

DC value:

Make this information visible on schematic: □

Small signal AC analysis(.AC)

AC Amplitude: 1

AC Phase: 0

Make this information visible on schematic: ☑

Parasitic Properties

Series Resistance[Ω]:

Parallel Capacitance[F]:

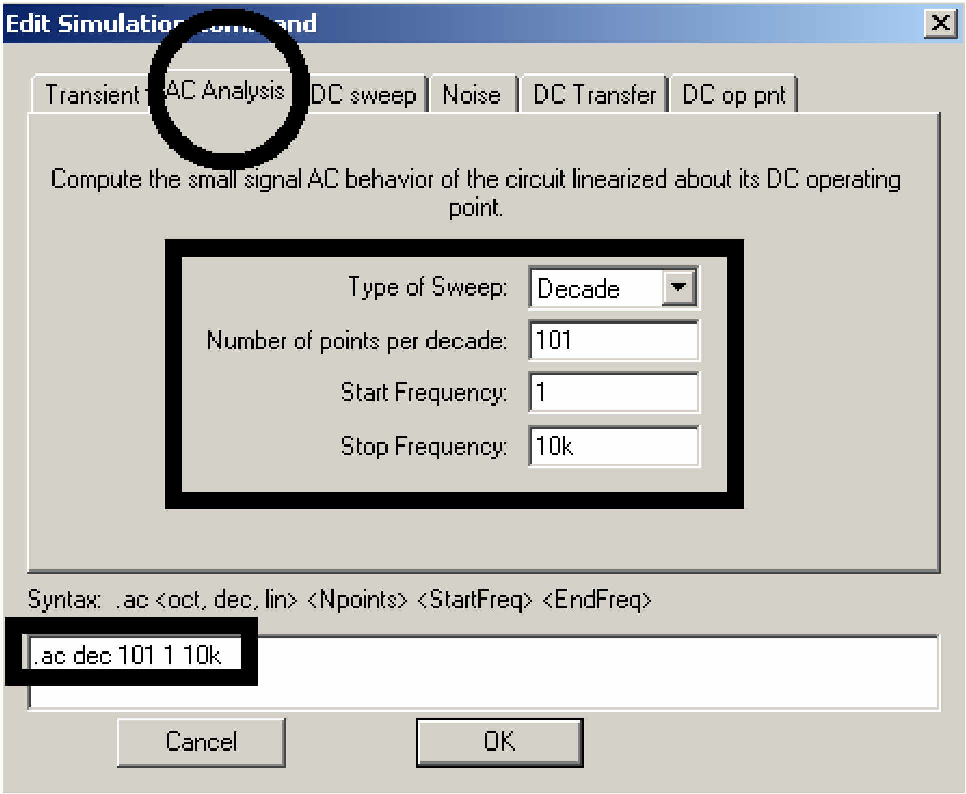
Make this information visible on schematic: ☑

Step 1: Switch the voltage source to „**AC Sweep**“ by opening the property menu and the two entries:

AC Amplitude = 1 Volt

AC Start Phase = 0

Ignore the programming for the time domain of the last example and delete the mark in the field „Make this information visible on schematic“.



Edit Simulation Command

Compute the small signal AC behavior of the circuit linearized about its DC operating point.

Type of Sweep: Decade

Number of points per decade: 101

Start Frequency: 1

Stop Frequency: 10k

Syntax: .ac <oct, dec, lin> <Npoints> <StartFreq> <EndFreq>

```
.ac dec 101 1 10k
```

Step 2: Open „**Simulate**“ and the line „**Edit Simulation Command**“.

Change to the „**AC Analysis**“ menu and enter:

Decade sweep

101 points per decade

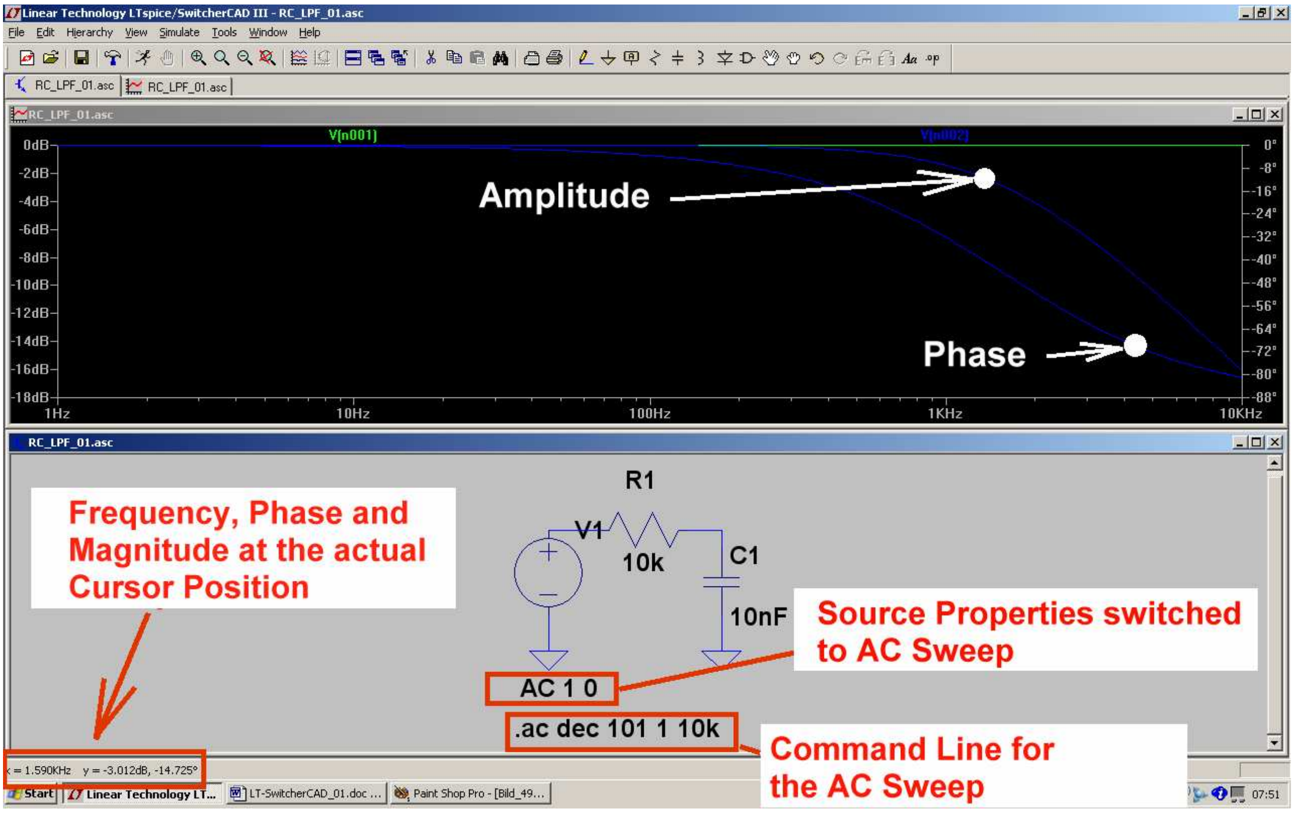
Start Frequency = 1Hz

Stop Frequency = 10kHz

Please check now the generated simulation command:

.ac dec 101 1 10k

Then start the simulation and test the result. Use the cursor to see the actual frequency, phase and magnitude.



Amplitude

Phase

Frequency, Phase and Magnitude at the actual Cursor Position

Source Properties switched to AC Sweep

Command Line for the AC Sweep

AC 1 0

.ac dec 101 1 10k

x = 1.590KHz y = -3.012dB, -14.725°

Congratulations, the first own example is now successfully terminated!

5. FFT (= Fast Fourier Transformation)

Very often we need information about all frequencies which can be found in a signal. It is a fact that only a pure sine wave consists of only one frequency. So, every deviation from the ideal sine wave curve produces new frequencies (harmonics) with frequency values of 2x, 3x, 4x….of the fundamental frequency. This information can be obtained by using the **FFT**.

But first several things are neeeded to get the FFT working and to get good results.

===

The start frequency AND the minimum width of a spectral line AND the smallest frequency step are all determined by the

Mimimum frequency step = start frequency = minimum line width =
= 1 / simulation time

Also for periodic signals the simulation time must be an integer multiple of the period time. If you ignore that you get “ghost lines“ and / or an additional noise floor.

(An additional note for specialists only: you should reduce the used simulation time by one timestep. Otherwise you are already starting a new period which will not be complete. But this error is only noticeable for very “coarse time steps” or for a very small number of periods).

==

Necessary properties of the **maximum time step**:

a) Do not make it too coarse. This avoids „corners“ in the simulated time domain signal.

b) The related „**minimum sample frequency = 1 / maximum time step“** must have a minimum frequency value twice of the maximum frequency line in the simulated signal. Otherwise the „Nyquist - Shannon – Law“ is violated and you get undesired aliasing effects.

c) The maximum frequency AND the dynamic range of a FFT are increased by the number of used samples in the FFT. But: these samples must be true and you must not use more samples for the FFT than you get from the time domain simulation. The number of samples can easily be calculated by

True number of samples = simulation time / maximum time step

==

LTspice uses data compression for the results file of the time domain simulation. This is undesirable because it looses informations and also a lot of the mentioned „true samples“.
So switch off the data compression by the following Spice directive:

```
.options    plotwinsize=0
```

===

For a FFT the number of used samples must always be a multiple of „2“.
No problem because LTspice will always default to those in the choice menu.
Note::
More „true“ samples give a higher stop frequency AND an extended amplitude range in the simulated spectrum. But the necessary calculation time AND the resultant data file also increases.
===
For specialists only.
Because only the maximum time step is fixed, the sample rate applied by SPICE can vary or be increased automatically when simulating “the difficult parts of a curve”. But this cannot be forseen, equates to an additional FM and causes additional but unexpected spectral lines and an increased “noise floor” at high frequencies.
This effect can be reduced by choosing the minimum sample rate as high as possible (= maximum time step as low as possible) and choosing as less samples for the FFT as possible.

Example:

Let us use the RC-LPF of the last chapter (R =10kΩ / C = 10nF). The input is fed by a symmetric square wave voltage (minimum amplitude value = 0V , maximum amplitude value = 1V , frequency = 1kHz).

If we use a simulation time of 20ms with a maximum time step of 100ns we meet the above recommendations as follows:

a) The simulation time of 20ms is an integer multiple of the signal's period time (1ms).

b) A maximum time step of 100ns is corresponding to a minimum sample frequency of 1/100ns = 10MHz. This gives a Shannon corner frequency of 0.5 x sample frequency = 5MHz.

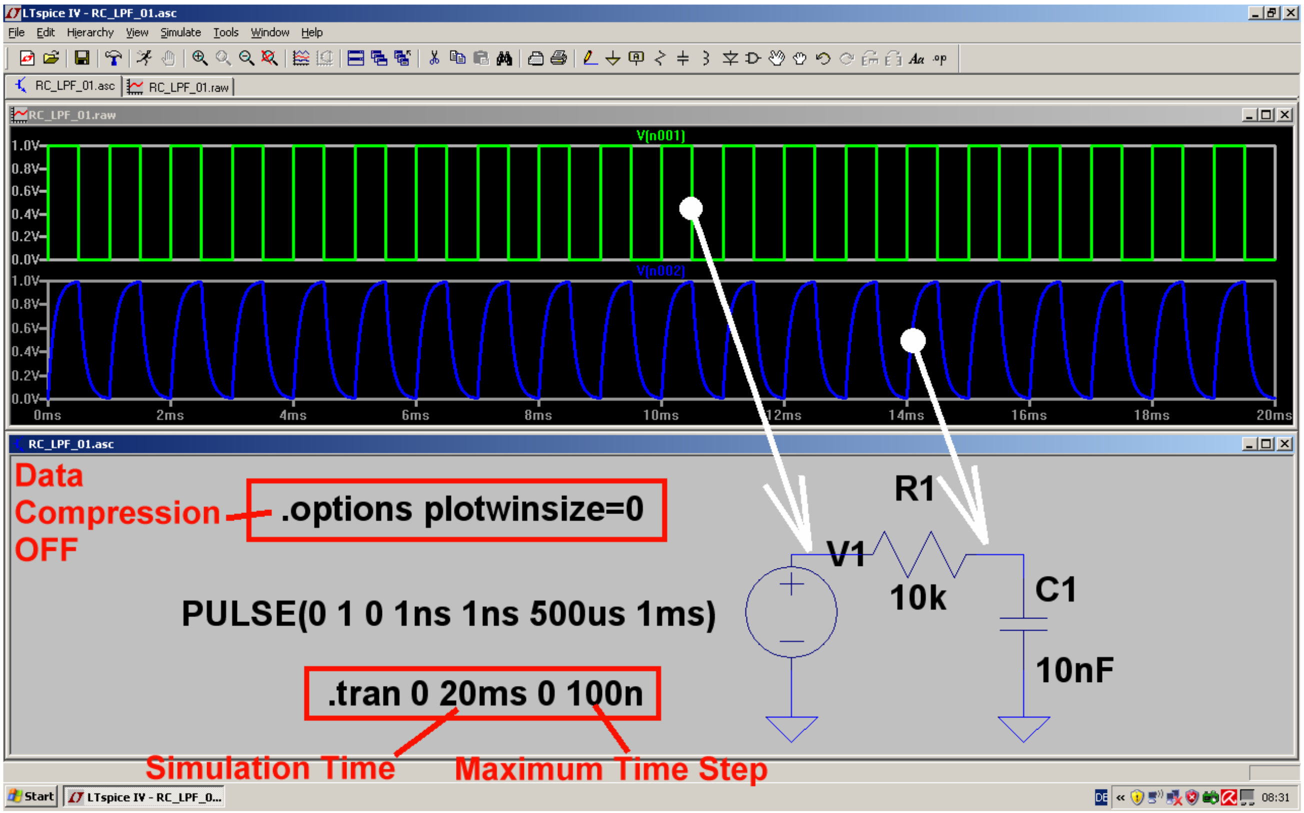
c) The number of true samples is **20ms / 100ns = 200 000**. So we enter

„131 072 sampled data points in time"

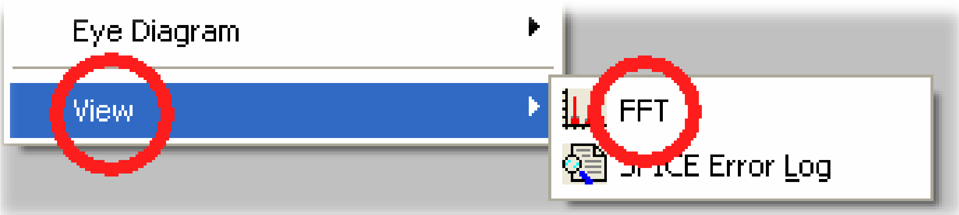
in the FFT menu.

Now draw the circuit, set the properties of the voltage source, choose simulation time and time step and write the directive to switch off the data compression.

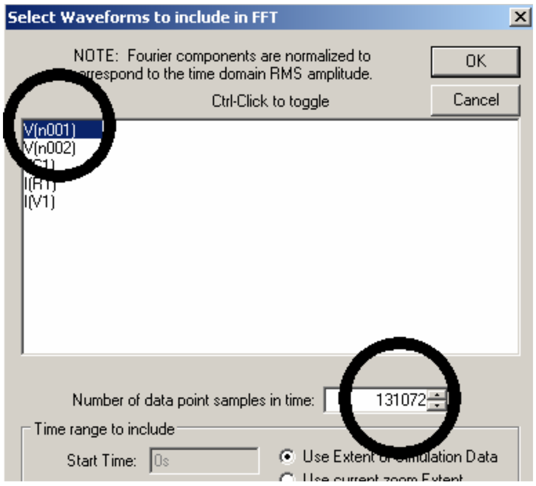
The time domain simulation result (= input and the output voltage) should be presented in two plot panes. Also "thick lines" are used for the curves (= button with the hammer / menu "waveforms").



```
.options plotwinsize=0
PULSE(0 1 0 1ns 1ns 500us 1ms)
.tran 0 20ms 0 100n
```



Now right click on the input voltage and choose "View" and "FFT"



Select Waveforms to include in FFT

NOTE: Fourier components are normalized to correspond to the time domain RMS amplitude.

Ctrl-Click to toggle

V(n001)
V(n002)
I(C1)
I(R1)
I(V1)

Number of data point samples in time: 131072

Time range to include

Start Time: 0s

● Use Extent of Simulation Data

○ Use current zoom Extent

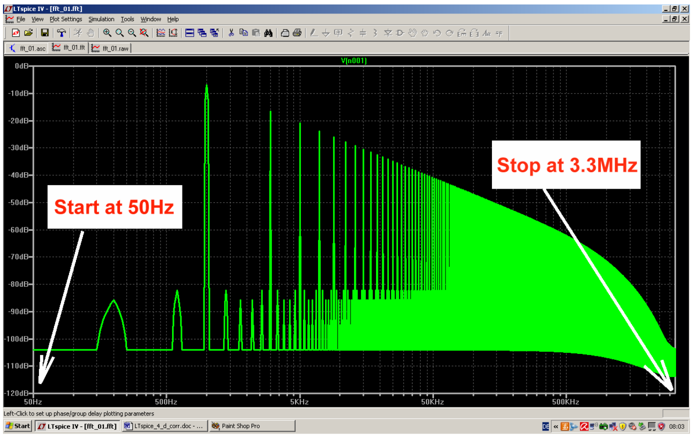
Select **Input Voltage V(n001)** by a mouseclick

As discussed we have 200 000 real samples of this curve. So we enter a

“number of 131 072 data point samples in time”

and start the FFT.

This is the successful result:



Start at 50Hz

Stop at 3.3MHz

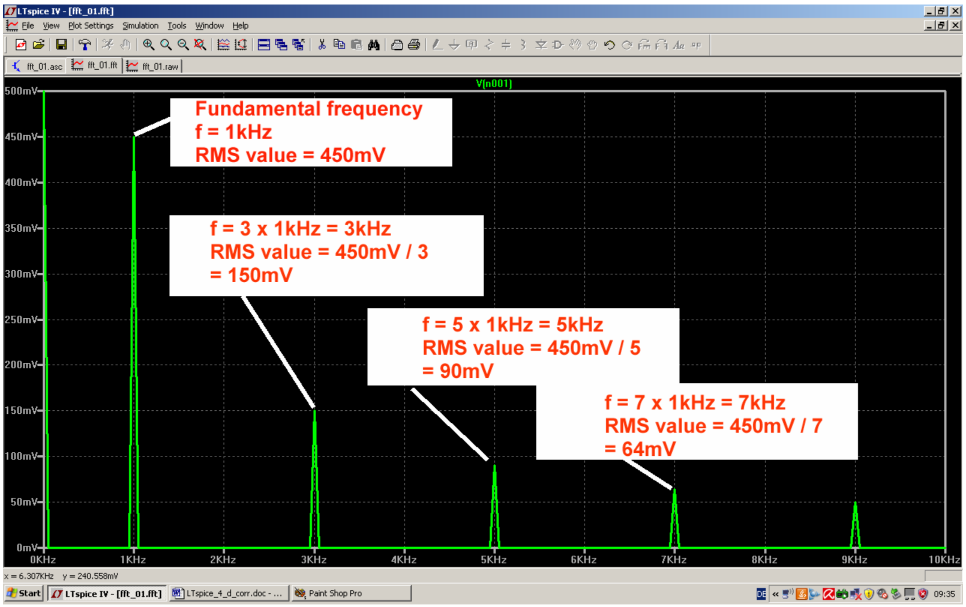
Start frequency = frequency resolution =

50Hz = 1 / 20ms.

The stop frequency is 3.3MHz and can only be increased by using a higher number of samples.

Now change the diagram to linear vertical and horizontal result presentation to confirm theory which says:

In a symmetric square wave you can only find the fundamental frequency and its odd harmonics. The amplitudes of the harmonics decrease exactly with their order:



Fundamental frequency
f = 1kHz
RMS value = 450mV

f = 3 x 1kHz = 3kHz
RMS value = 450mV / 3
= 150mV

f = 5 x 1kHz = 5kHz
RMS value = 450mV / 5
= 90mV

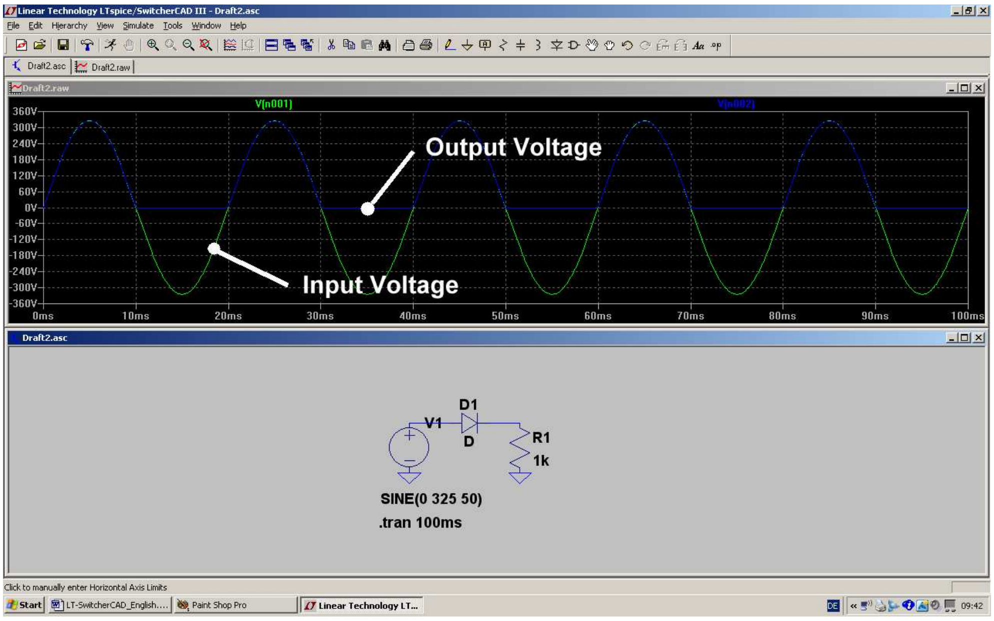
f = 7 x 1kHz = 7kHz
RMS value = 450mV / 7
= 64mV

Note:
If anybody is additionally interested in the phase information: click on the scaling of the right vertical diagram axis and press “Phase” in the menu.

6. Project 2: Rectifiers

6.1. Simple Rectifier without Transformer

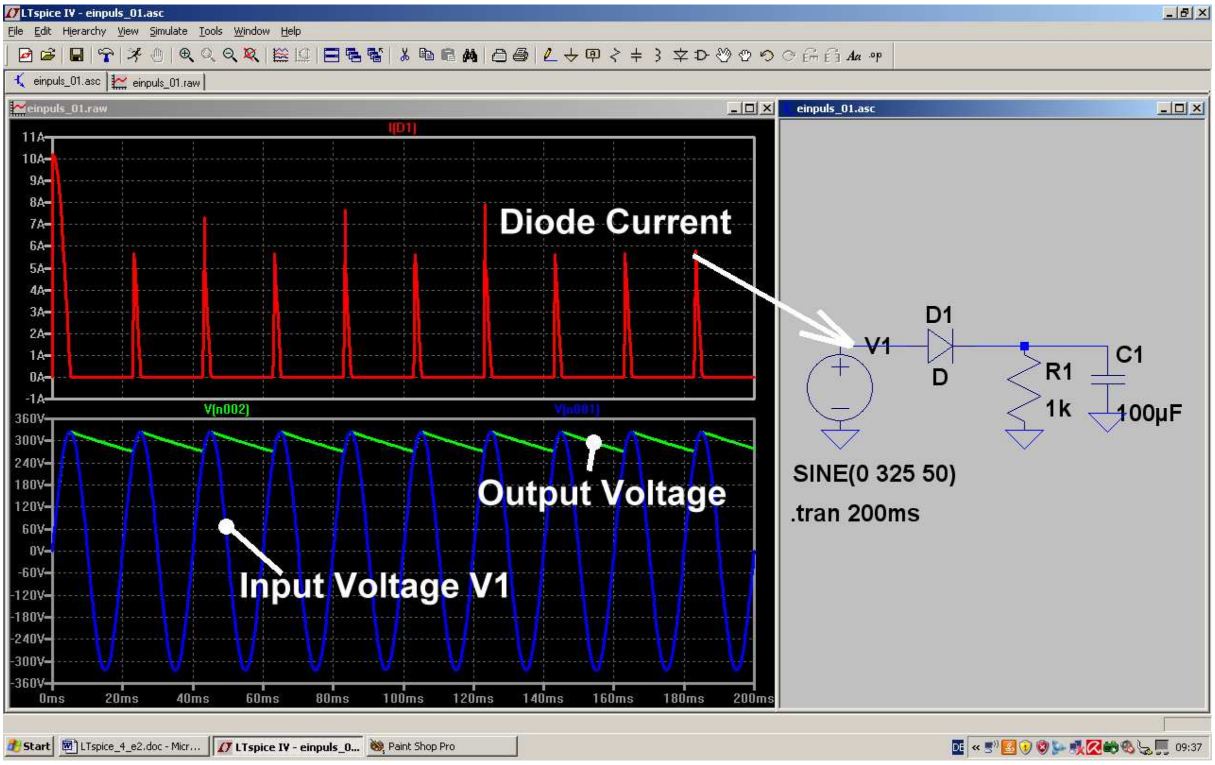
No problem: draw a circuit consisting of a Voltage Source, a diode and a resistor (1k). Now apply a sine wave (50 Hz / peak value = 325 V) to the input and simulate from 0 to 100ms.



```
SINE(0 325 50)
.tran 100ms
```

Task:
Add a parallel capacitor of 100 Mikrofarad to the output.
Show also the diode current (by moving the cursor over the diode’s symbol to change the cursor into a current sensor).

Solution:



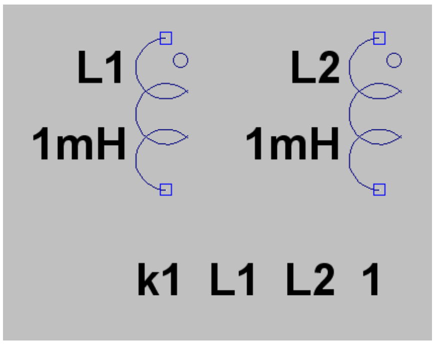
```
SINE(0 325 50)
.tran 200ms
```

6.2. Important: creating SPICE Model and Symbol for a Transformer

6.2.1. The easiest Solution: a simple ideal Transformer

Sorry, but in the LTSpice part library you can't find any transformer! So let us start with a simple but good solution found in the original SPICE manual:

Place the necessary windings (with their Inductances) in the schematic and add a SPICE Directive for the magnetic coupling "kn" between these windings!



```
k1 L1 L2 1
```

Example for a two winding transformer and voltage transfer ratio = 1:1

a) Place "**Ind2**" from the symbol library two times on the screen, because only at this symbol the start of the winding procedure is marked by a circle.

b) The line "**k1 L1 L2 1**" says that the **magnetic coupling factor between the windings L1 and L2 is "1" = 100%.**

c) Attention: **It is not possible to enter a given voltage transfer ratio "(N1 / N2)" into PSPICE!!**

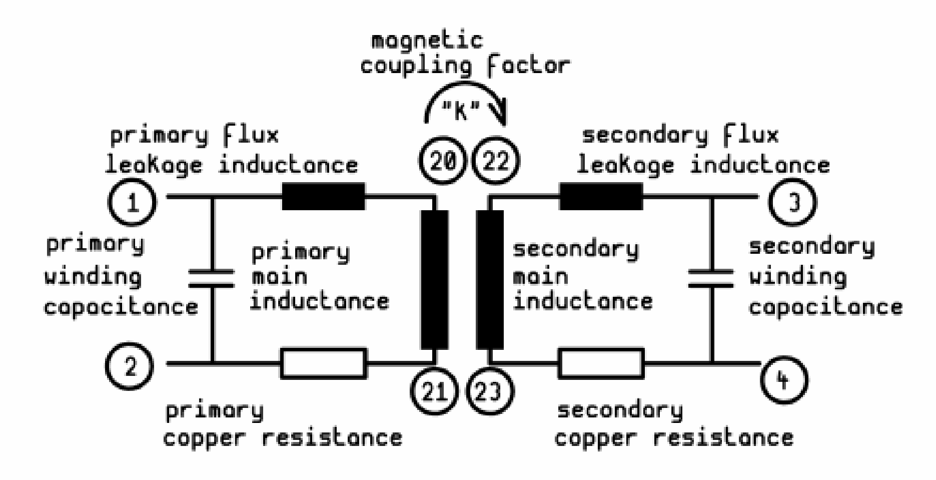
The reason is not complicated: PSPICE works always and only with parts and their properties (here: inductances and magnetic coupling factor). So we must use the inductances and the following relation:

$$\frac{L_{Primary}}{L_{Secondary}} = \left(\frac{N1}{N2}\right)^2$$

So choose a realistic value for L1 and then calculate L2 by using this formula.

6.2.2. Creation of the SPICE Model for a real Two Windings Transformer

We use the following transformer model for our work, which includes the „magnetic coupling" between the primary and secondary winding, the flux leakage inductances, the winding capacitances and the copper resistors of the windings.



Node 1 and 2 are the primary winding connections, node 3 and 4 the connections for the secondary winding of the model for the simulation. For the "inner nodes" you should choose much higher numbers to avoid collisions when "blowing up the circuit diagram" with more windings ore parts.

Now have a look at the following SPICE – Model, using the **subcircuit „xformer_01“ with its connections 1 / 2 / 3 / 4:**

(Lleak is the name for the **flux leakage inductances** which represent the part of the flux who does not run through both main inductances leaded by the magnetic core of the transformer. All other circuit elements correspond to the above diagram):

```
*
*   [1]   ---.  ||
*            )  ||   .---   [3]
*            )  ||  (
*            )  ||  (
*   [2]   ---.  ||   .---   [4]
*

.SUBCKT xformer_01  1  2  3  4
*
** Primary
Lleak1     1        20      1mH
Lpri1      20        21     1H
Rpri1      21        2      1
Cpri1      1         2      20pF
*
* *Secondary
Lleak3     3         22     1mH
Lsec1      22       23      1H
Rsec1      23       4       1
Csec1       3       4       20pF

K   Lpri1 Lsec1     0.999
.ENDS
```

Explanation:
Every line starts with the name of the part, followed by the numbers of the nodes between this part is inserted in the circuit. And at last you find the parts value.

In praxis the magnetic coupling is always below „1“ (due to the flux leakage). So always insert a value smaller than 1 (here: k = 0.999). Some SPICE versions start a protest action if you don’t follow this advice.

Primary and secondary winding use the same number of turns. So you have to enter the same inductance value to get a 1:1 - transformer.

Note:
The description of the circuit starts always with **.SUBCKT ……………** and ends with **.ENDS**

===
Very important:
a) The finished model is named **xformer_01.lib** an should be saved in the folder

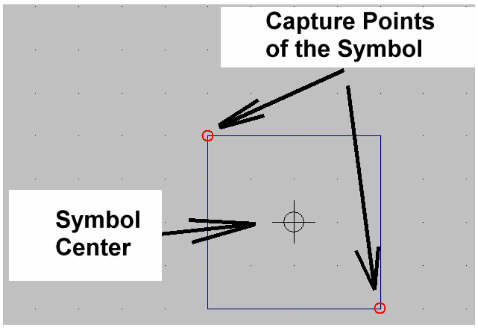
Programs / LTC / LTspiceIV / lib / sub

where all other LTC-SPICE-models are already located. Please pay attention to the correct path and the correct extension when saving this file..….

b) It **is impossible** to enter the turn ratio! The only possibility is to enter different primary and secondary main inductances and we’ll handle this in the next project.
===

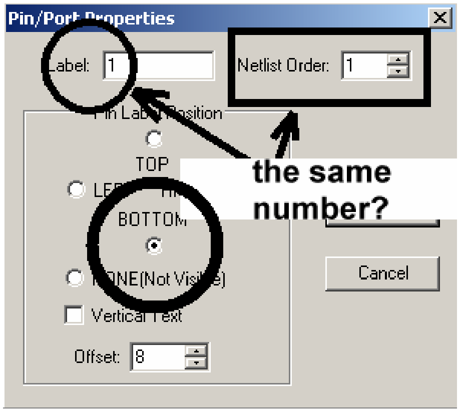
6.2.2. Creation of the Symbol for a Two Windings Transformer

Step 1:
Open „**File**“ and click on „**New Symbol**“. Now you see an empty screen with “cross and circle” in its center.



Step 2:
Open „**DRAW“**, choose „Rect“ (= rectangle) and draw it with the dimensions 20mm x 20mm around the symbol’s center point (…as a help: You find a fine point grid with 5mm distance on the screen).

(The red circles at two corners of the rectangle are „capture points“ for the user when placing this symbol in a circuit).



Pin/Port Properties

Label: 1

Netlist Order: 1

Pin Label Position

○ TOP

○ LEFT ○ RIGHT

● BOTTOM

○ NONE(Not Visible)

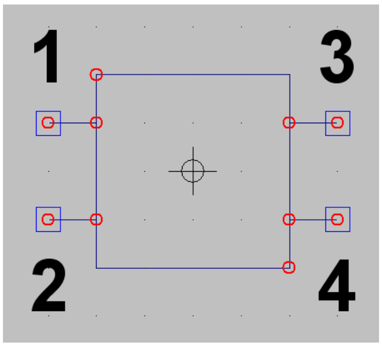
□ Vertical Text

Offset: 8

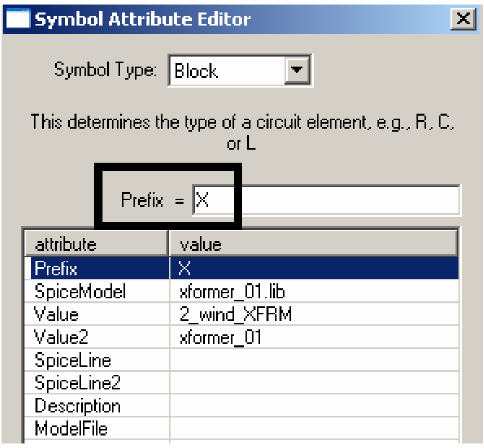
Step 3:
Go to „**Edit**“ and „**Add Pin / Port**“ to generate the connection points. But in the menu you have at every connection point not only to enter the pin Number BUT ALSO the relative position between this point and the Point’s Label.

And always examine carefully whether the point’s label and netlist order are identical!

Then click OK and you have the pin at your cursor. Now you can place it in a distance of 5mm from the symbol (See next figure).



Step 4:
Generate all 4 Pins and afterwards open once more „**DRAW**“ to connect the pins to the symbol by wires.
So the terminated symbol should look like.



Symbol Attribute Editor

Symbol Type: Block

This determines the type of a circuit element, e.g., R, C, or L

Prefix = X

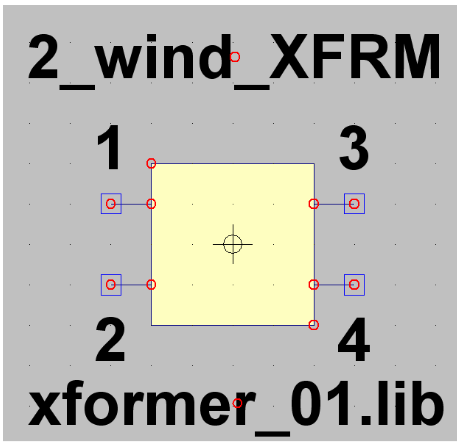
attribute	value
Prefix	X
SpiceModel	xformer_01.lib
Value	2_wind_XFRM
Value2	xformer_01
SpiceLine	
SpiceLine2	
Description	
ModelFile	

Step 5:
Let us complete the real transformer for the simulation. Please open „**EDIT / Attributes / Edit Attributes**“ to arrive at this menu.

Now enter all entries as given in the figure.

(**Remark: Prefix „X“ means a subcircuit).**

Warning:
The line for „Value2“ MUST NEVER BE EMPTY --- otherwise you can get a cryptic error message! Normally use the same entry as in the line for the spice model, but without extension.



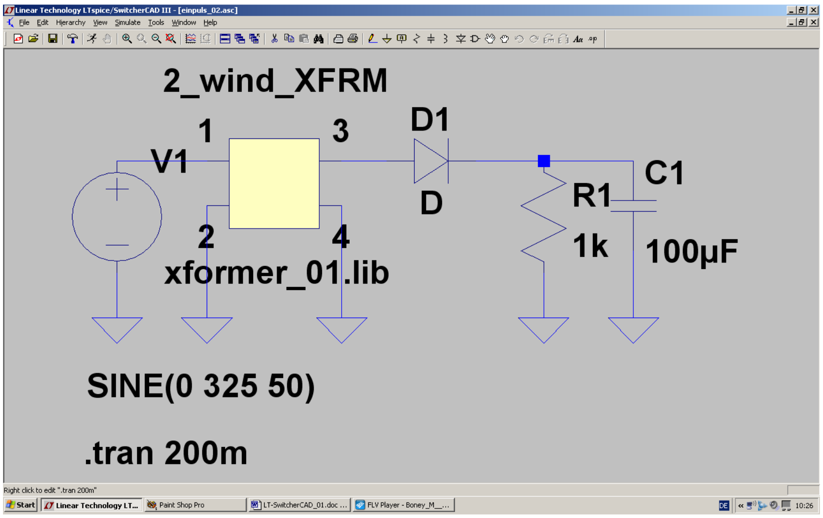
Step 6:
At last make visible all important informations for the user.
Start with „**Edit**“ and „**Attributes**“, but now choose „**Attribut Window**“. When clicking on „**value**“, then you get the description „**2_wind_XFRM**“ at your cursor and you can place it over the symbol.
Then repeat the procedure but now choose „**SpiceModel**“ = **xformer_01.lib** and place it below the symbol. The work is done!

Step 7:
Save the complete symbol.
Open „**File**“ and „**Save as**“ and the correct path starting at **„SwCADIII“ and „lib“ just down to the folder „sym“ (= symbols)**. There create a new folder named „**Xformers**“ and save this new part as **„xformer_01.asy**“.

6.3. One Pulse Rectifier with Transformer

Now let us test a circuit with line transformer, one pulse-rectifier and output capacitor.

Task:
Test a power supply consisting of a line transformer, rectifier, load resistor R1 = 1kΩ and load capacitor C1 = 100µF. Line input voltage is 230Vrms / 50 Hz, the turn ratio of the transformer is 20:1. Simulate in the time domain from 0...200ms.


Linear Technology LTspice/SwitcherCAD III - [einpuls_02.asc]
File Edit Hierarchy View Simulate Tools Window Help
2_wind_XFRM
1
3
D1
V1
D
R1
1k
C1
100µF
2
4
xformer_01.lib
Start
Linear Technology LT...
Paint Shop Pro
LT-SwitcherCAD_01.doc ...
FLV Player - Boney_M_...
10:26

SINE(0 325 50)

.tran 200m

Right click to edit ".tran 200m"

Solution:
Most of the necessary parts are already known from the last example. Fetch and place them, wire and enter the correct part values including the properties of the voltage source and add the command for the simulation from 0 to 200ms in the time domain.

But now we need the line transformer of the last chapter. Fetch and place the part.

Then we look for the necessary transformer property entries.

You cannot enter the turn ratio N1/N2 in a direct form, so you need the following equation for the inductances:

$$\frac{L_{primary}}{L_{secondary}} = \left(\frac{N_1}{N_2}\right)^2$$

Let us apply a value of 1Henry for the primary main inductance (….ideal transformer…) and calculate the secondary main inductance for a turn ratio of 20:1 in the following manner:

$$L_{secondary} = \frac{L_{primary}}{(20)^2} = \frac{1H}{400} = 2.5mH$$

```
*
* [1]        ---. ||
*               ) || .---    [3]
*               ) || (
*               ) || (
* [2]        ---. || .---    [4]
*

.SUBCKT xformer_01 1 2 3 4
*
* Primary
*
Lleak1  1       20      1uH
Lpri1   20      21      1H
Rpri1   21      2       1
Cpri1   1       2       1pF

* Secondary
*
Lleak3  3       22      1uH
Lsec1   22      23      2.5mH
Rsec1   23      4       1
Csec1    3      4       1pF

K   Lpri1 Lsec1           0.999
.ENDS
```

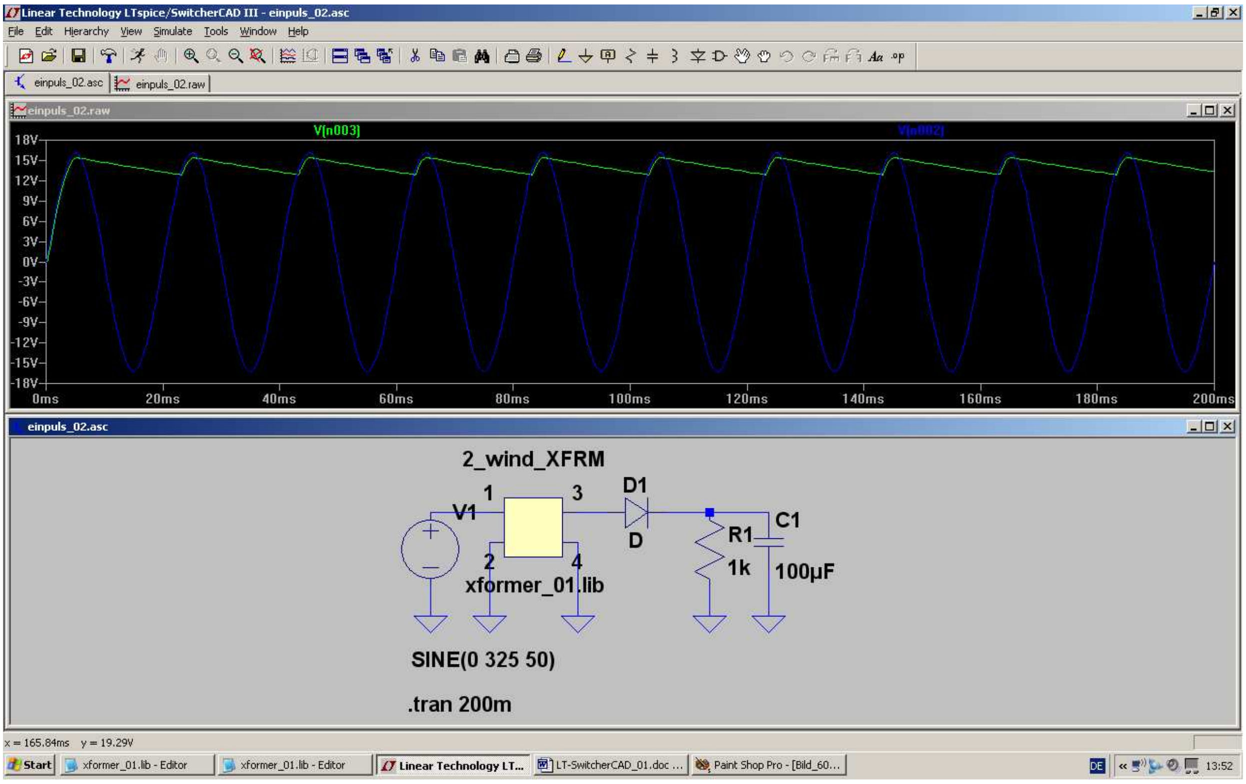
For an ideal transformer you can reduce the values for the flux leakage inductances, the copper resistors and the winding capacitances.

But always you must never set them to zero”......

Especially a resistor of zero Ohms in series to an inductance causes in a “normal SPICE version” at once an error message or a simulation abort!

So this model file of the transformer is a good template for your work. Open the „xformer_01.lib“ – File, change the entries and save it.

Now the simulation runs and here you have the result screen with the voltages at the left and the right side of the diode:



SINE(0 325 50)

.tran 200m

6.4. Rectifier with Diode 1N4007

We want to use the well known universal line rectifier diode **1N4007** instead of the ideal diode.
Sorry; but the SPICE-model of this type cannot be found in the LTC-library…
So let us tackle the import of foreign models (from the WEB.....).

Step 1:

Use Google and look for **diode.lib**
(This is an ORCAD-library with a lot of rectifiers).

But be carefully:
You get this library as a HTML-file and you must not use it in this form!!!!!

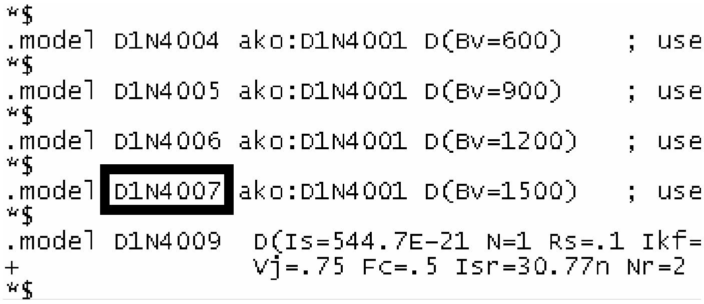
So mark the content by **<STRG> + <a>**, copy to the clipboard by **<STRG> + <c>** and insert it into a new file of a text editor (like Notepad).

Now save this file as „**diode.lib**“ in the folder

LTC / LTspiceIV / lib / sub

(But pay attention to get the correct extension „*.lib“ by choosing the type “**all files**”. Otherwise the editor will add a further extension “*.txt” and the library cannot be found by our program…**)**

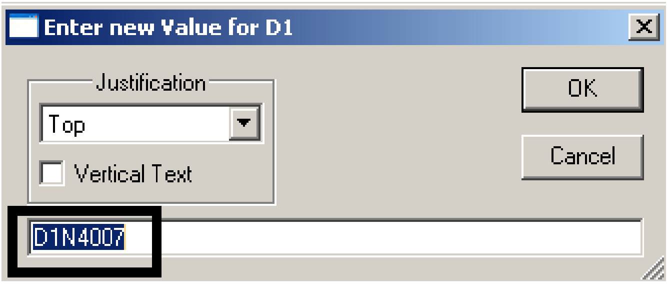
Step 2:



```
*$
.model D1N4004 ako:D1N4001 D(Bv=600)    ; use
*$
.model D1N4005 ako:D1N4001 D(Bv=900)    ; use
*$
.model D1N4006 ako:D1N4001 D(Bv=1200)   ; use
*$
.model D1N4007 ako:D1N4001 D(Bv=1500)   ; use
*$
.model D1N4009  D(Is=544.7E-21 N=1 Rs=.1 Ikf=
+               Vj=.75 Fc=.5 Isr=30.77n Nr=2
*$
```

Now open „diode.lib“ with the text editor and search for „**1N4007**“. You will then recognize that the correct diode model name is

D1N4007



Enter new Value for D1

Justification: Top

☐ Vertical Text

D1N4007

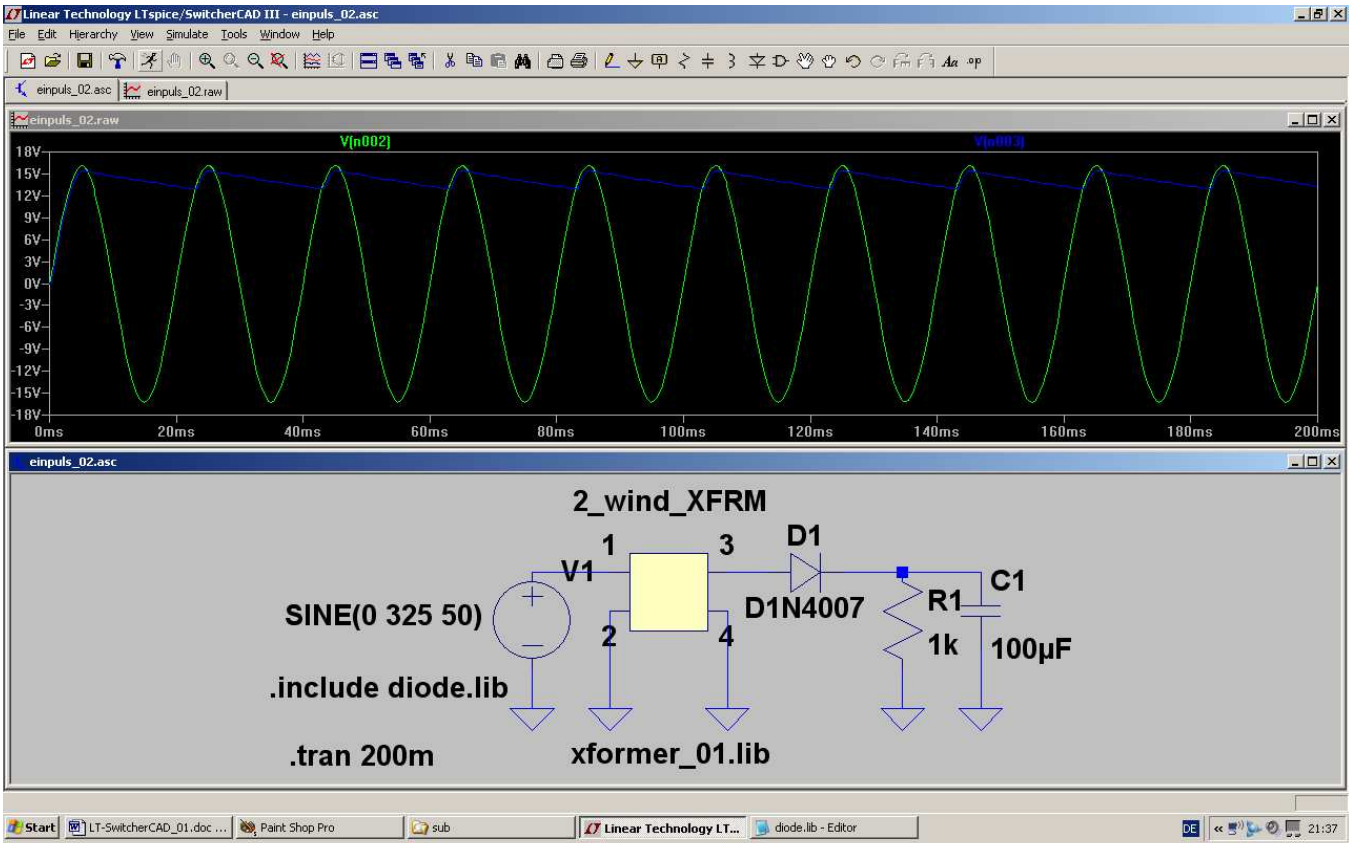
Step 3:
Move the cursor on the letter „D“ at the symbol of the diode and right click with your mouse.

Then enter the new correct value „**D1N4007**“ and close by pressing OK.

Step 4:
Enable the program to find out the correction location of this D1N4007 model and add the following Spice directive on the screen:

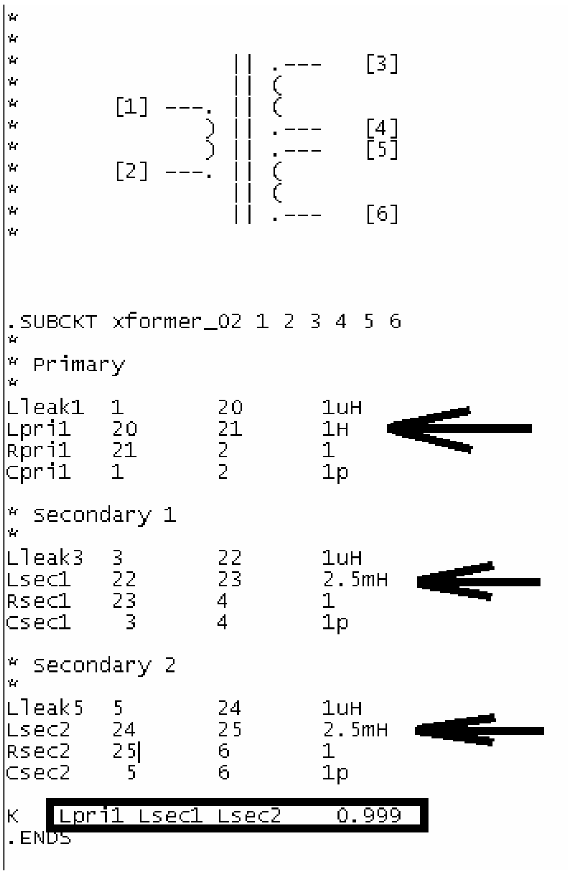
.include diode.lib

Step 5: self-explainig….



6.5. Two Pulse Rectifier with real Transformer

Here we need at first something new: a line transformer with 2 secondary windings.



```
*
*
*                                || .---    [3]
*                                || (
*         [1] ---.               || (
*                )               || .---    [4]
*                )               || .---    [5]
*         [2] ---.               || (
*                                || (
*                                || .---    [6]
*


.SUBCKT xformer_02 1 2 3 4 5 6
*
* Primary
*
Lleak1  1       20      1uH
Lpri1   20      21      1H
Rpri1   21      2       1
Cpri1   1       2       1p

* Secondary 1
*
Lleak3  3       22      1uH
Lsec1   22      23      2.5mH
Rsec1   23      4       1
Csec1    3      4       1p

* Secondary 2
*
Lleak5  5       24      1uH
Lsec2   24      25      2.5mH
Rsec2   25      6       1
Csec2    5      6       1p

K   Lpri1 Lsec1 Lsec2     0.999
.ENDS
```

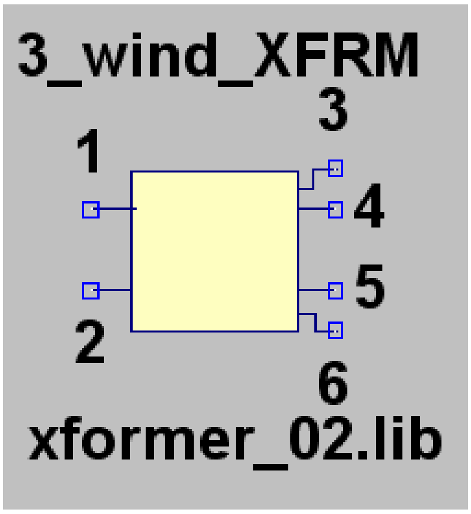
Step 1:
We open our well-known file „**xformer_01.lib**“ to modify it. Let the turn ratio be again 20:1 and add an additional secondary winding, also with 2.5mH. Write the necessary lines in your file (...they are a simple copy of secondary winding 1, but be aware of the node names...)

Reduce the flux leakage inductances and the copper resistors to get an ideal transformer, but don't set them to zero.

Please do not forget:
No we have 3 windings which are magnetally coupled. You have to enter a new coupling information

K   Lpri1 Lsec1 Lsec2 0.999

At last save the file as „**xformer_02.lib**“ in the “sub“-folder of the library.

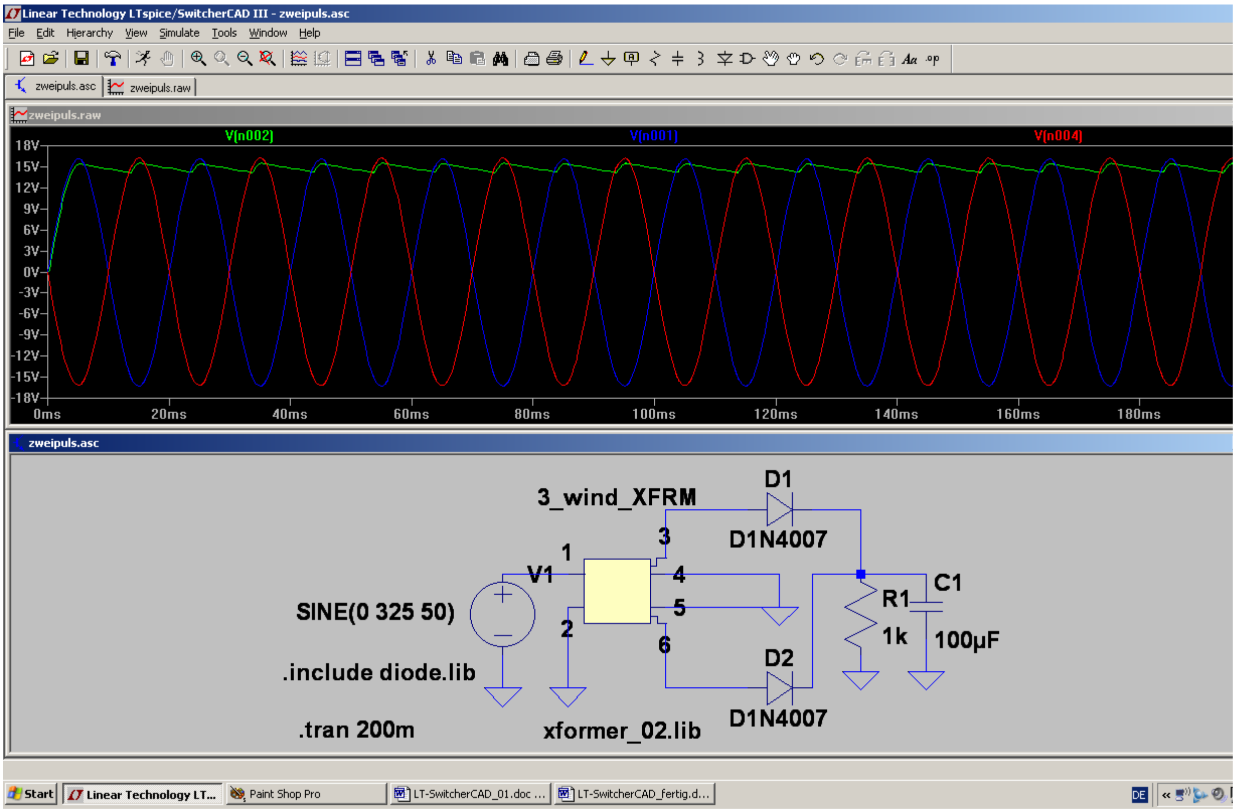


Step 2:
Also a new symbol is necessary and must be created -- see Chapter 6.2.2.

But do not forget:

Warning:
The line for Value2 MUSS NEVER BE EMPTY....otherwise you can get a cryptic error message and / or a simulation abort...

Step 3:
Draw the circuit using the new transformer and two diodes of the 1N4007-type. Then simulate and compare the result with the following page:

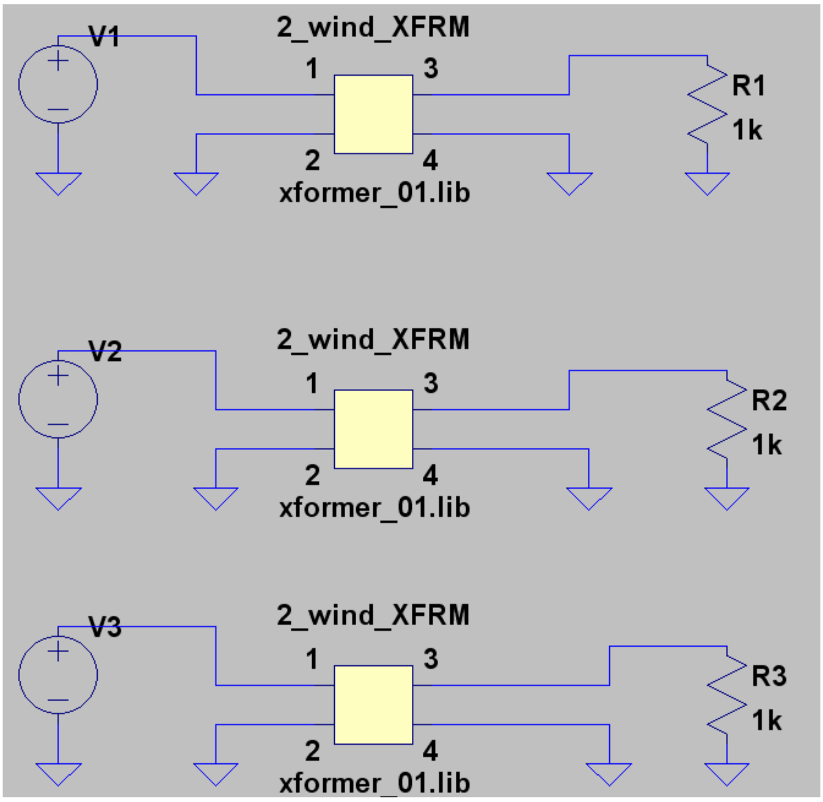


Linear Technology LTspice/SwitcherCAD III - zweipuls.asc
File Edit Hierarchy View Simulate Tools Window Help
zweipuls.asc
zweipuls.raw
V(n002)
V(n001)
V(n004)
18V
15V
12V
9V
6V
3V
0V
-3V
-6V
-9V
-12V
-15V
-18V
0ms
20ms
40ms
60ms
80ms
100ms
120ms
140ms
160ms
180ms
3_wind_XFRM
D1
D1N4007
1
2
3
4
5
6
V1
SINE(0 325 50)
.include diode.lib
.tran 200m
xformer_02.lib
R1
1k
C1
100µF
D2
D1N4007
Start
Linear Technology LT...
Paint Shop Pro
LT-SwitcherCAD_01.doc ...
LT-SwitcherCAD_fertig.d...
DE

----

7. Project 3: Three Phase AC System

7.1. Schematic and Simulation

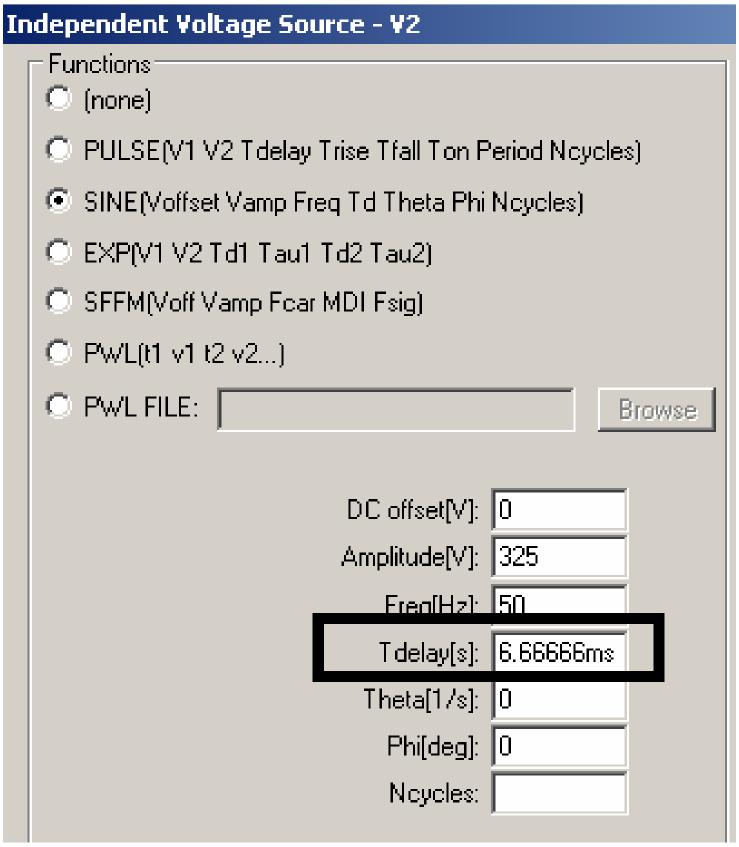


We need 3 voltage sources, 3 transformers and 3 load resistors to review this technique. For the transformers we will use our well-known file

„xformer_01.lib“,

but in this case **choose identical main inductances on the primary and secondary side to get a turns ratio of 1:1.**

Fetch all these parts from the library and draw the circuit diagram.


Independent Voltage Source - V2
Functions
○ (none)
○ PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)
● SINE(Voffset Vamp Freq Td Theta Phi Ncycles)
○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)
○ SFFM(Voff Vamp Fcar MDI Fsig)
○ PWL(t1 v1 t2 v2...)
○ PWL FILE:
Browse

DC offset[V]: 0

Amplitude[V]: 325

Freq[Hz]: 50

Tdelay[s]: 6.66666ms

Theta[1/s]: 0

Phi[deg]: 0

Ncycles:

Then move the cursor over the symbol of every voltage source and right click. All entries for the 3 sources are identical (except one....):

DC offset = 0 Volt

Amplitude = 325 V (peak value for RMS = 230V)

Frequency = 50 Hz

Phi = 0 degrees (start phase of the sine curve)

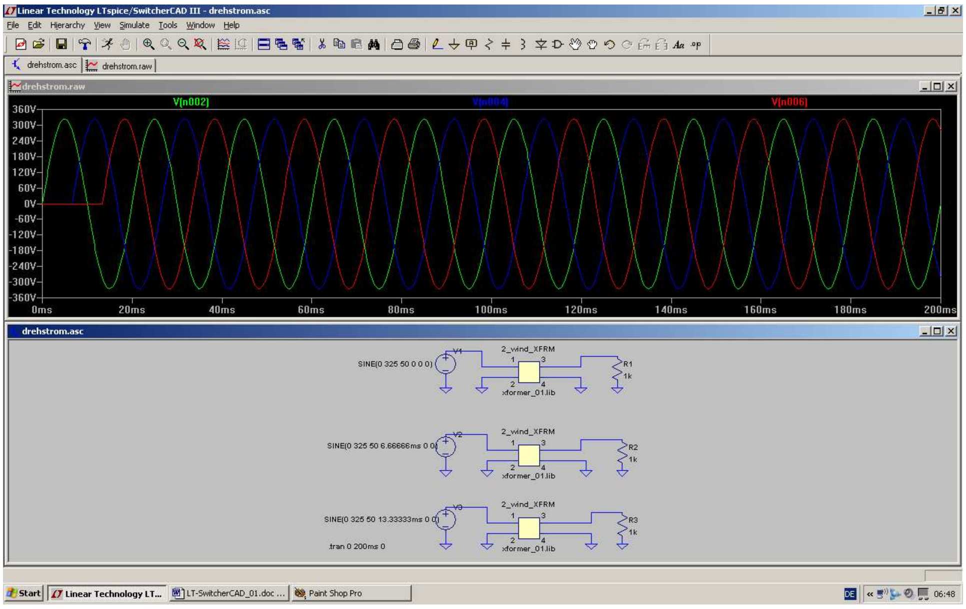
Important:
The 3 voltages have the same amplitude and the same frequency, but differ in the phase shifts. We cannot enter the necessary phase difference of 120 degrees in the property menu -- so simply enter different start delays:

for V1 the delay time is 0ms

for V2 the delay time is 6,66666 ms (see figure)

for V3 the delay time is 13,33333ms

When simulating for 200ms, you will get the following result:



SINE(0 325 50 0 0 0)

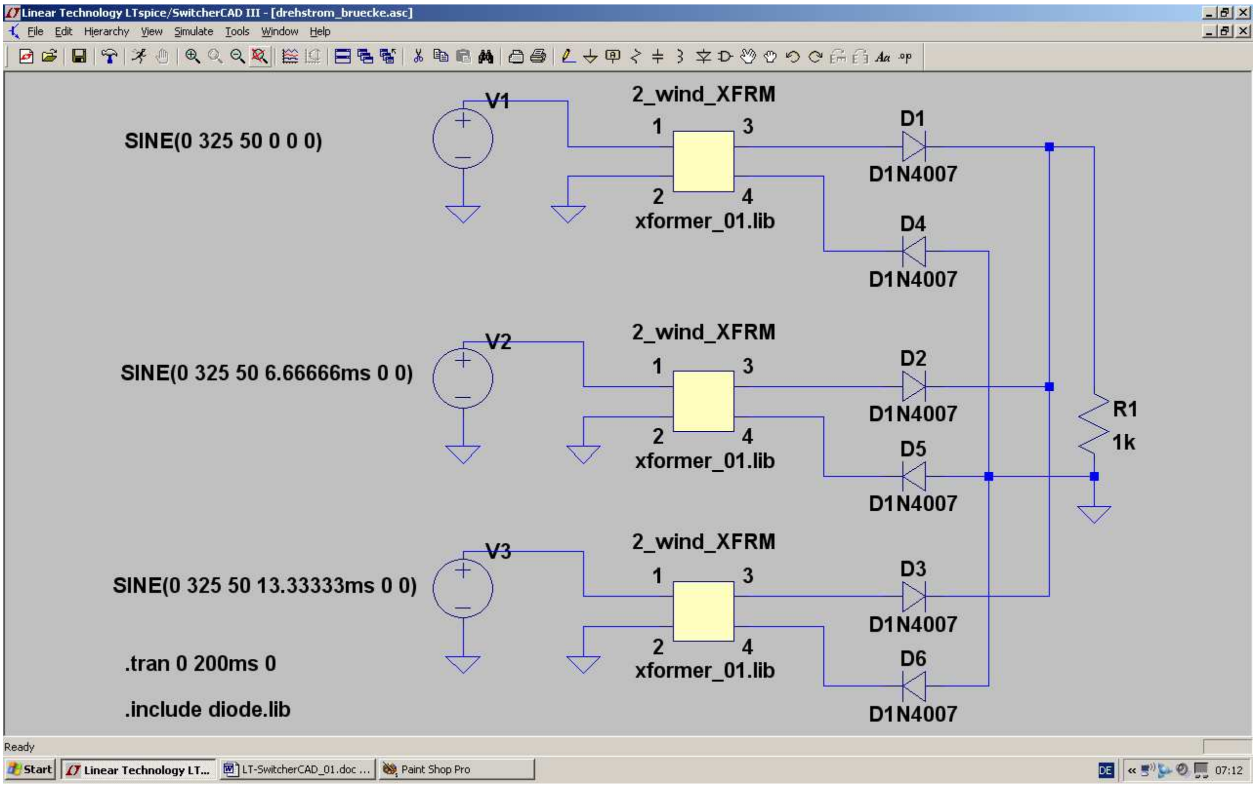
SINE(0 325 50 6.66666ms 0 0)

SINE(0 325 50 13.33333ms 0 0)

.tran 0 200ms 0

7.2. Rectifier Bridge for Three Phase AC Systems (e. g. Dynamo in a modern Car or Motorcycle)

Save this circuit in a new project folder and give it an appropriate name. Then modify the circuit by adding **6 diodes (1N4007)**. Delete two of the three resistors and wire the circuit as shown:


Linear Technology LTspice/SwitcherCAD III - [drehstrom_bruecke.asc]
File Edit Hierarchy View Simulate Tools Window Help
V1
V2
V3
2_wind_XFRM
1
3
2
4
xformer_01.lib
D1
D2
D3
D4
D5
D6
D1N4007
R1
1k
Ready
Start
Linear Technology LT...
LT-SwitcherCAD_01.doc ...
Paint Shop Pro
DE
07:12

SINE(0 325 50 0 0 0)

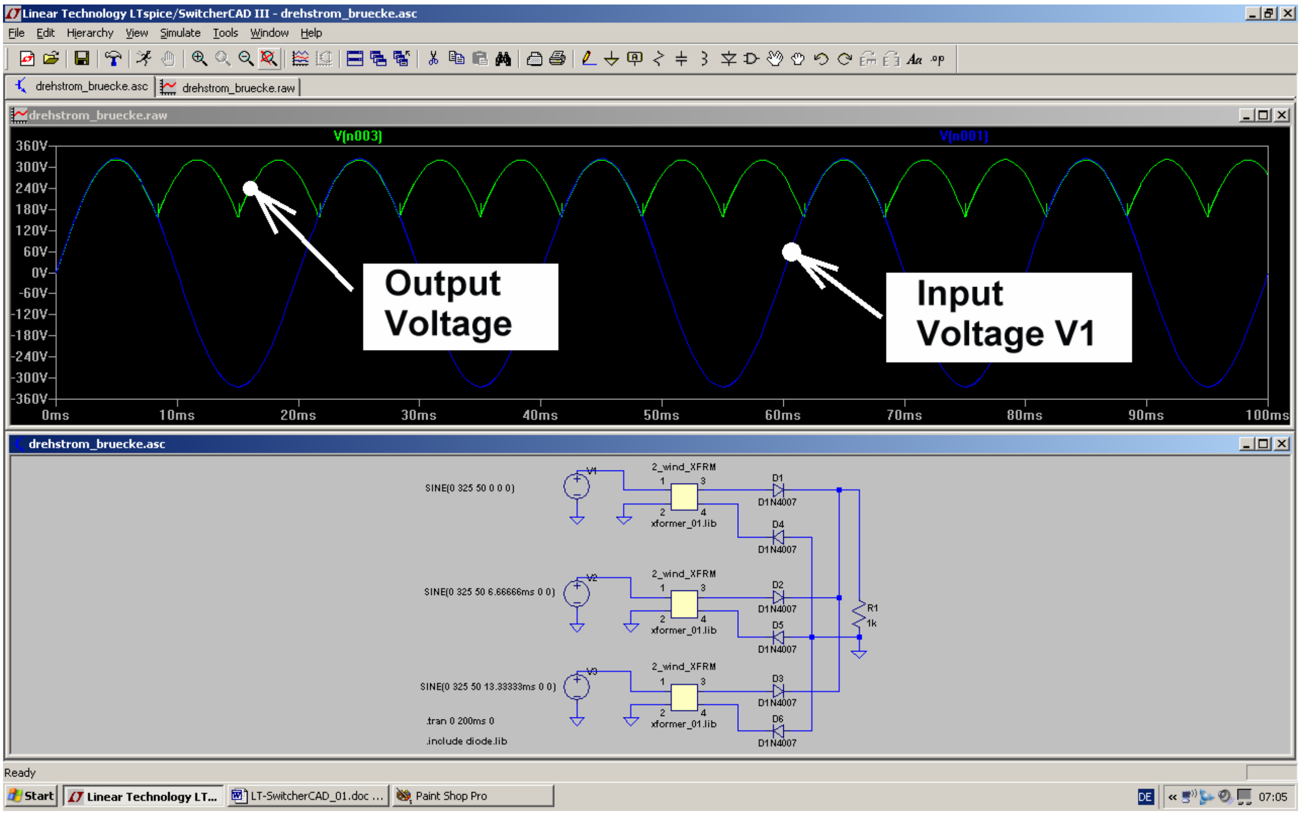
SINE(0 325 50 6.66666ms 0 0)

SINE(0 325 50 13.33333ms 0 0)

.tran 0 200ms 0

.include diode.lib

If you now simulate, you will get the following result for the input voltage V1 and the output voltage at R1:



Output Voltage

Input Voltage V1

SINE(0 325 50 0 0 0)

SINE(0 325 50 6.66666ms 0 0)

SINE(0 325 50 13.33333ms 0 0)

.tran 0 200ms 0

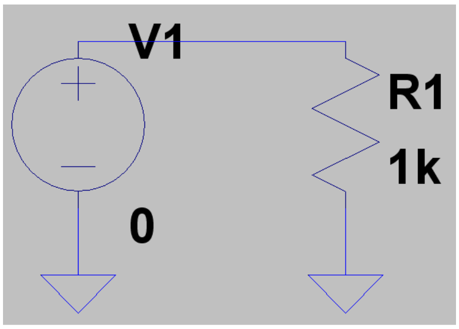
.include diode.lib

---

8. Project 4: V-I-Curves of Parts

8.1. Resistor

A resistor is the best object to make visible the V-I curve of a part.

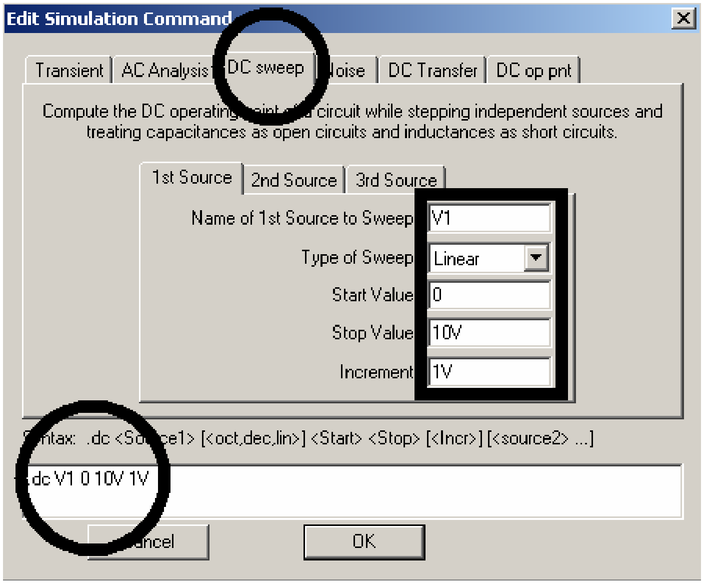


So open a new project and draw the schematic.

For the resistor R1 choose a value of 1kΩ.

Then open the property menu of the voltage source (by a right mouseclick on the "+"-sign in the symbol) and enter in the right upper corner of the menu "0" (= Zero Volts) as DC-Value.

Check the schematic for its correctness.



Compute the DC operating point of a circuit while stepping independent sources and treating capacitances as open circuits and inductances as short circuits.

Name of 1st Source to Sweep: V1

Type of Sweep: Linear

Start Value: 0

Stop Value: 10V

Increment: 1V

Syntax: .dc <Source1> [<oct,dec,lin>] <Start> <Stop> [<Incr>] [<source2> ...]

.dc V1 0 10V 1V

In the simulation menu open

„Edit Simulation Command"

and fill in the following DC-Sweep-entries for the voltage source V1:

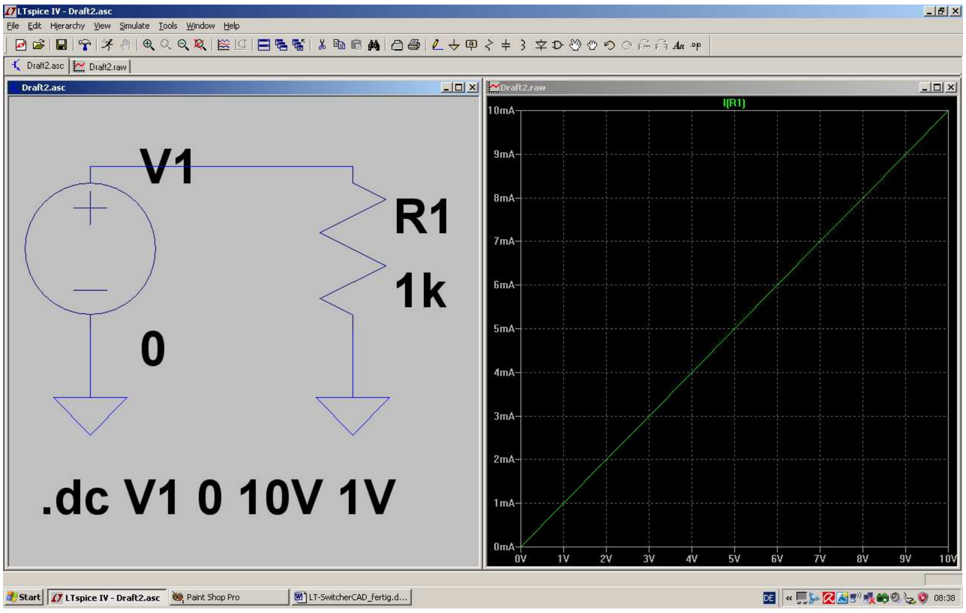
Start Value = 0 Volts

Stop Value = 10 Volts

Increment = 1Volt

Linear Sweep

Please check the so generated simulation command **„.dc V1 0 10V 1V "** in the lower part of the window.



Then close with OK and start the simulation. Move the cursor to the upper connection of the resistor R1. When the cursor suddenly looks like a "current sensor": left click and you get this screen.

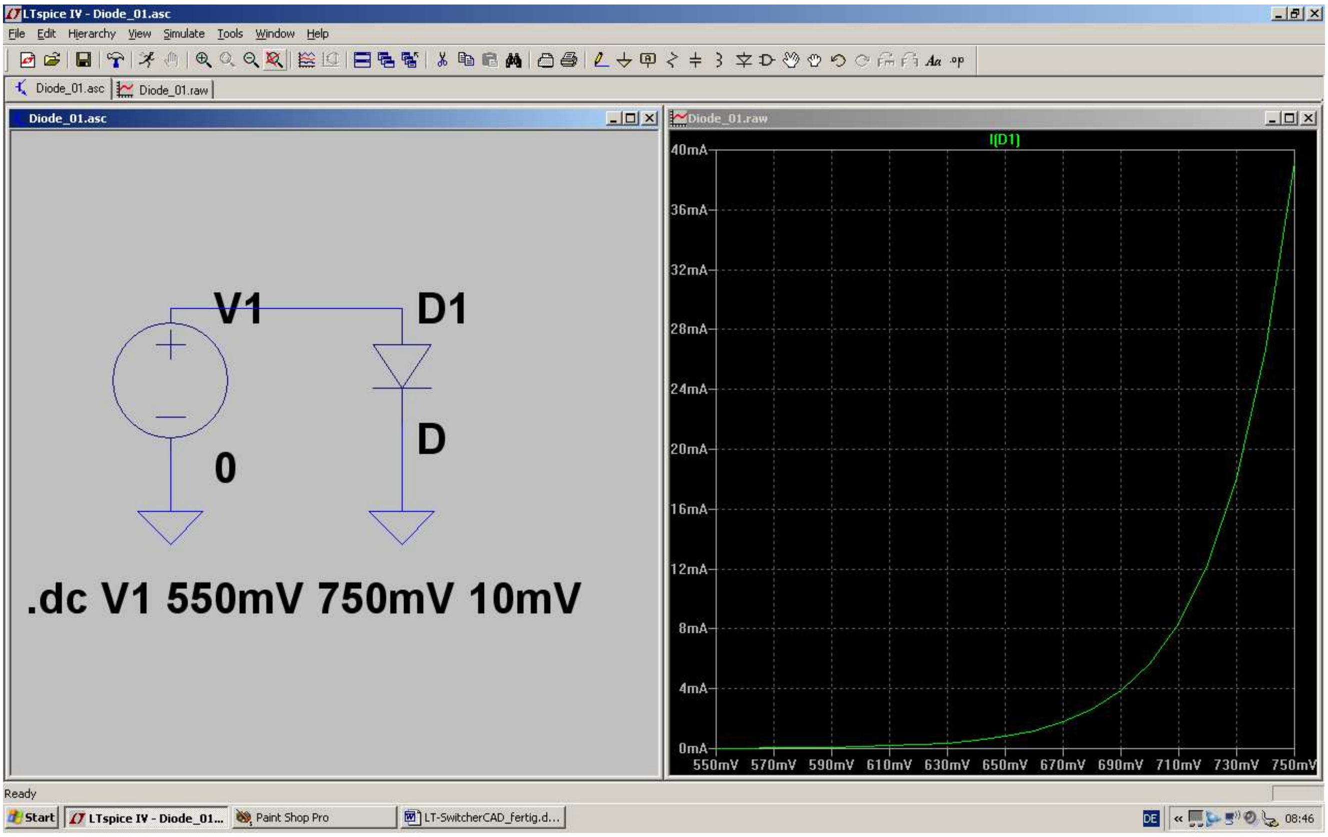
8.2. Diode

This is now a very simple exercise:

First save the project under a new name. Then delete the resistor (key F5) and replace it by a diode (= part „diode“ in the Library).
At last modify the simulation command to

.dc V1 550mV 750mV 10mV

Now the voltage range is 550mV to 750mV with an increment of 10mV. This is the optimum for a silicon diode.

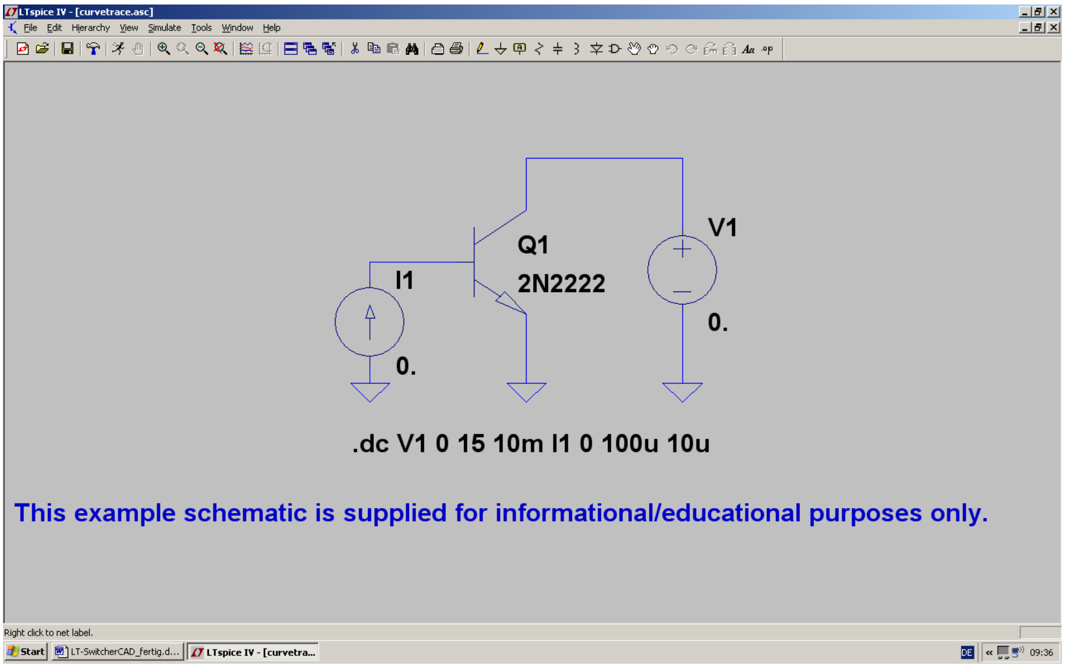


LTspice IV - Diode_01.asc
File Edit Hierarchy View Simulate Tools Window Help
Diode_01.asc
Diode_01.raw
Diode_01.asc
V1
D1
D
0
Diode_01.raw
I(D1)
40mA
36mA
32mA
28mA
24mA
20mA
16mA
12mA
8mA
4mA
0mA
550mV 570mV 590mV 610mV 630mV 650mV 670mV 690mV 710mV 730mV 750mV
Ready
Start
LTspice IV - Diode_01...
Paint Shop Pro
LT-SwitcherCAD_fertig.d...
DE
08:46

.dc V1 550mV 750mV 10mV

8.3. NPN Transistor

In the „example“ folder of the program you will find a very nice example „**curvetrace.asc**“ for this purpose:


LTspice IV - [curvetrace.asc]
File Edit Hierarchy View Simulate Tools Window Help
Q1
2N2222
I1
V1
0.
0.

.dc V1 0 15 10m I1 0 100u 10u

This example schematic is supplied for informational/educational purposes only.

Right click to net label.

Start
LT-SwitcherCAD_fertig.d...
LTspice IV - [curvetra...
DE
09:36

You can see:

a) A DC voltage source V1 is connected to the collector of the transistor. The voltage start value is “0”.

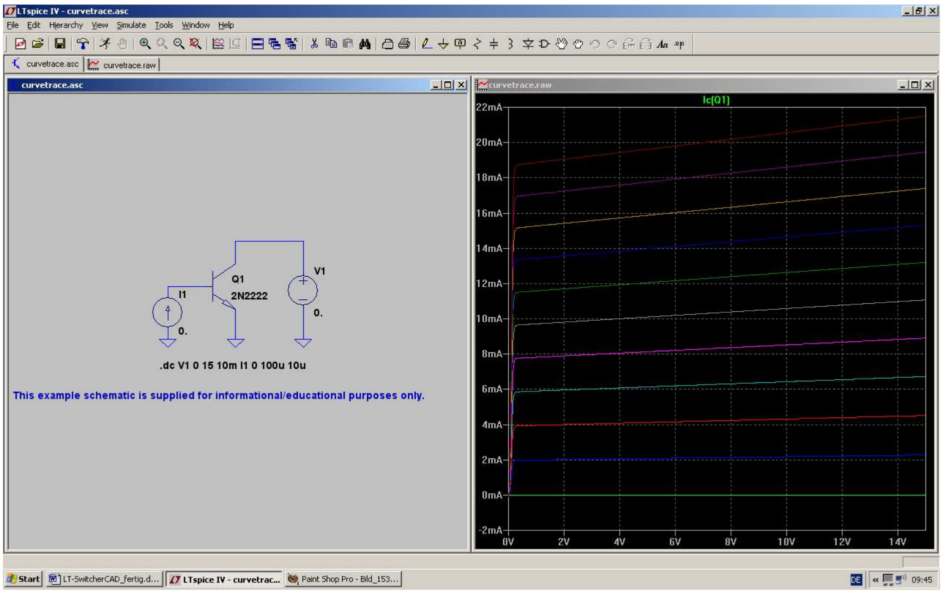
b) A current source I1 with start value „0“ feeds the base of the transistor.

The simulation command

.dc V1 0 15 10mV I1 0 100u 10u

means:

Increment voltage V1 from 0 Volts to +15 Volts in steps of 10mV. Use this voltage for the horizontal axis in the waveform viewer .
Then apply a base current between 0 and 100 Microamperes (incremented by steps of 10 Microamperes) and write the simulated curves into the waveform viewer.

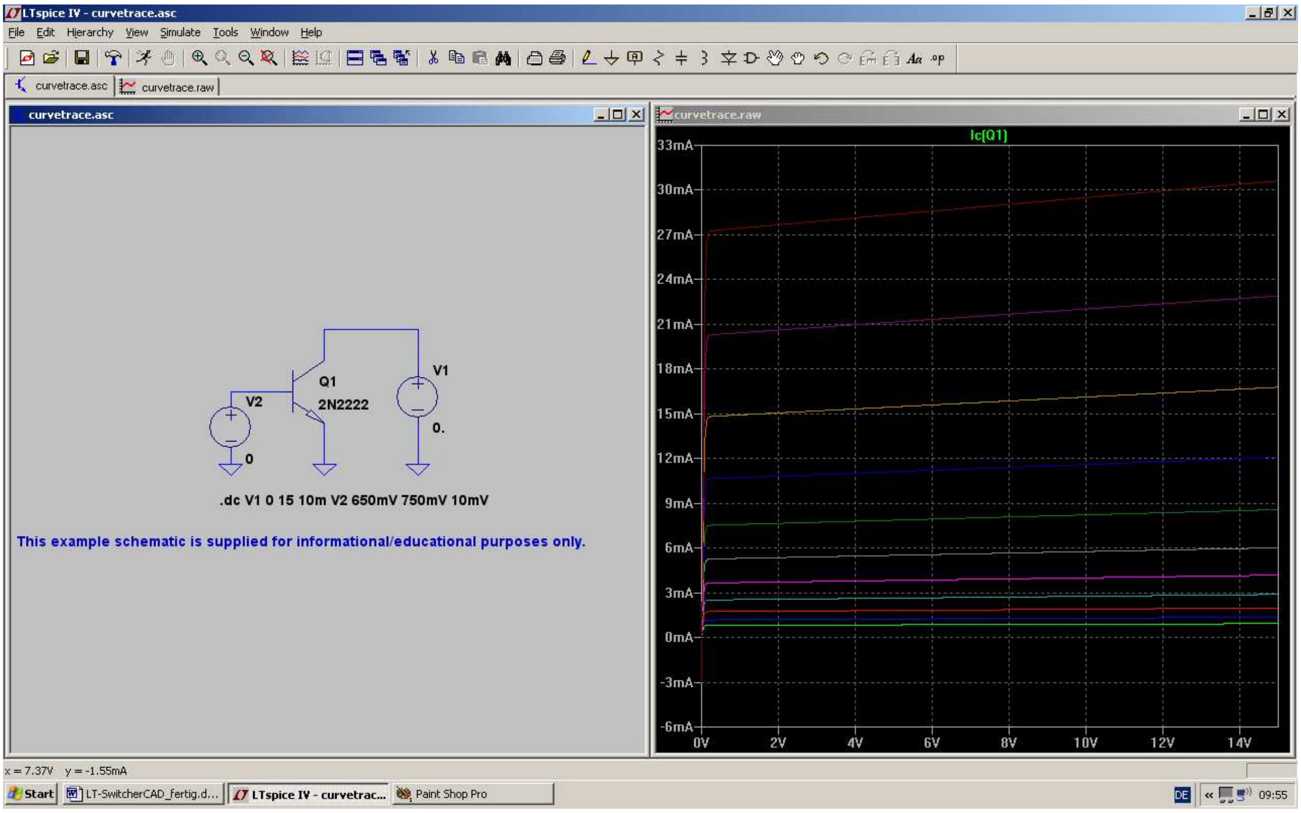

LTspice IV - curvetrace.asc
File Edit Hierarchy View Simulate Tools Window Help
curvetrace.asc
curvetrace.raw
curvetrace.asc
Q1
2N2222
I1
V1
0.
0.
curvetrace.raw
Ic(Q1)
22mA
20mA
18mA
16mA
14mA
12mA
10mA
8mA
6mA
4mA
2mA
0mA
-2mA
0V
2V
4V
6V
8V
10V
12V
14V
Start
LT-SwitcherCAD_fertig.d...
LTspice IV - curvetrac...
Paint Shop Pro - Bild_153...
DE
09:45

.dc V1 0 15 10m I1 0 100u 10u

This example schematic is supplied for informational/educational purposes only.

Move the cursor to the collector of transistor Q1. When the cursor suddenly looks like a “current sensor”, left click and you will get this screen.

If you are interested in the relation between collector current and base voltage, then you need this schematic:



.dc V1 0 15 10m V2 650mV 750mV 10mV

This example schematic is supplied for informational/educational purposes only.

x = 7.37V y = -1.55mA

a) Replace the base current source with a voltage source. Start value is “0”.

b) Modify the simulation command to:

.dc V1 0 15V 10mV V2 650mV 750mV 10mV

The voltage source V1 stays the same. The base voltage V2 is stepped from 650mV to 750mV by increments of 10mV.

That is all. See the result in the waveform viewer on the top right of this page....

8.4. N-Channel Junction FET

First we have to replace the NPNTransistor of the last example by the FET (= part „njf“ in the part library of the program). To show the example to its best we will use the well known

BF245B

First search on the internet with the Google entry

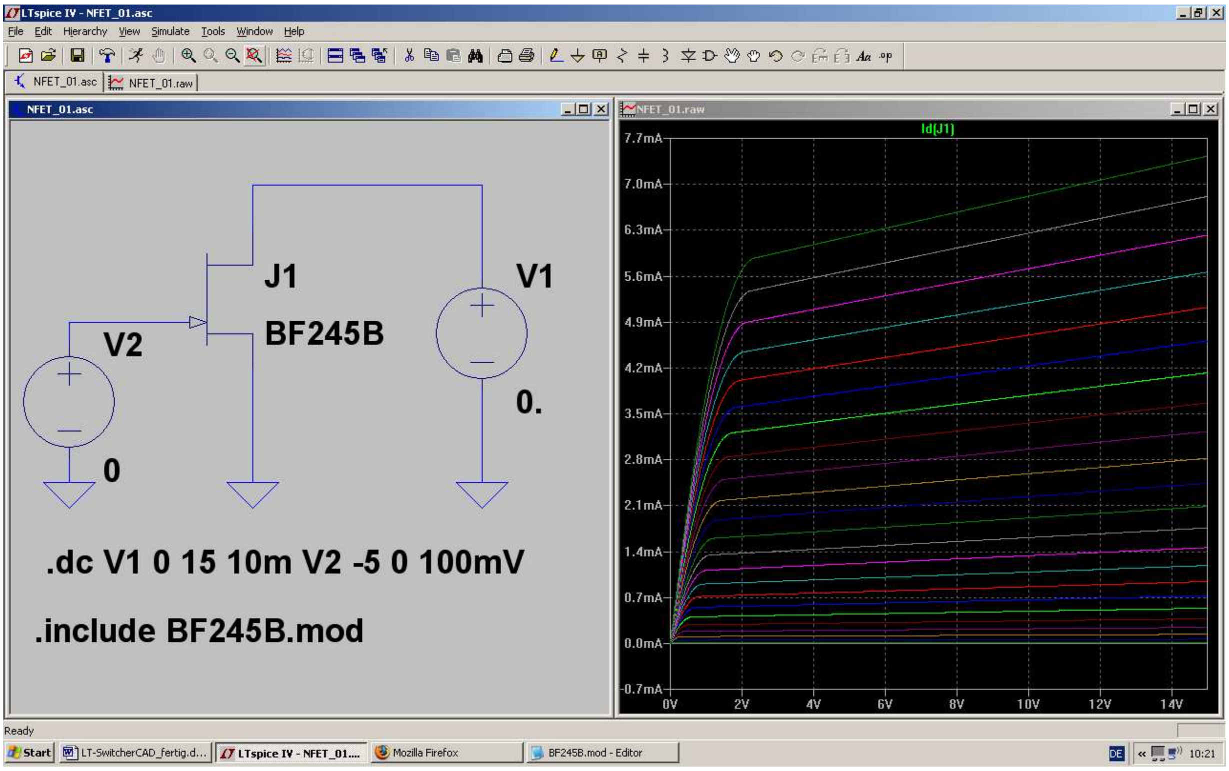
bf245b spice model

Copy the content of the result to the clipboard and then into a new editor sheet. Save this sheet as

BF245B.mod

in the folder **„sub“** of the program’s part library.

And then do the following:



```
.dc V1 0 15 10m V2 -5 0 100mV

.include BF245B.mod
```

First select the N-Channel Junction FET symbol from the library (= „njf“) and place it on the schematic. Then right click on the model description „NJF“ beside the symbol. Change it to **BF245B** and finally add a SPICE directive to the schematic:

.include BF245B.mod

Now the software will use the downloaded BF245B model for the simulation.

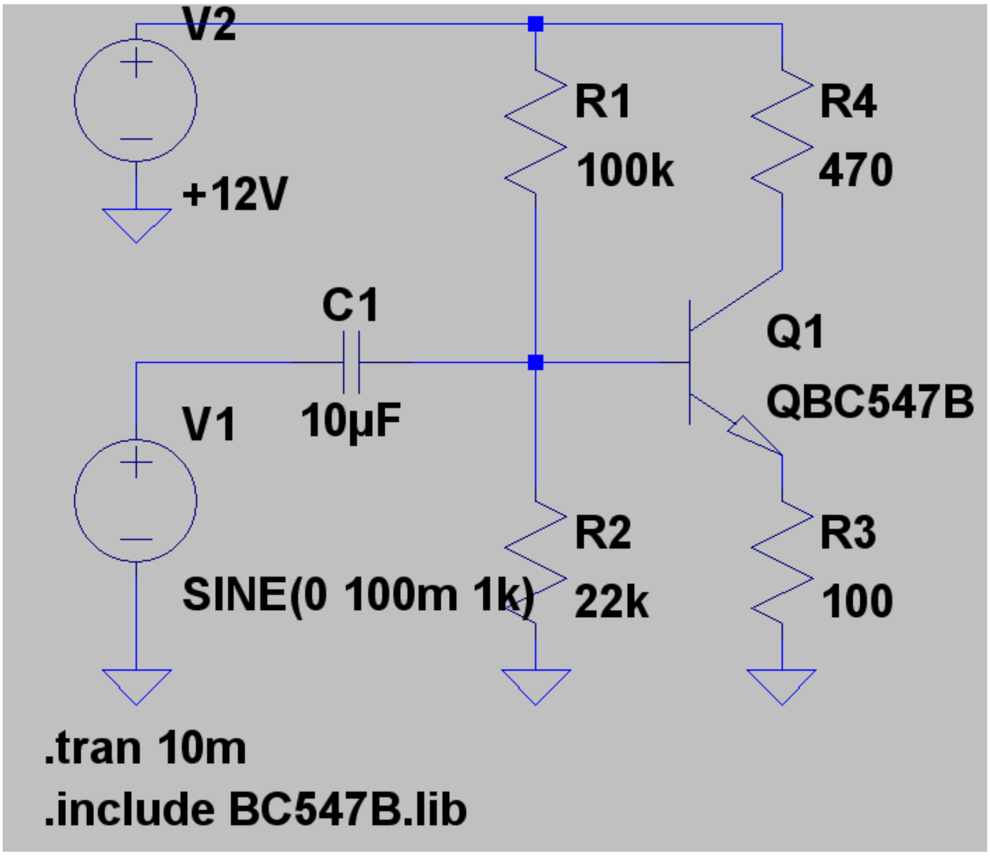
Lastly the gate voltage range must be altered. For a normal junction FET it is the best to use a range from

-5V to 0V in Steps of 100mV

The simulation should now run without any problems.

9. Project 5: Transistor circuits
9.1. One Stage Amplifier

9.1.1. Time Domain Simulation



We need a power supply, a sine voltage source, one capacitor, four resistors and an npn-transistor.

Let’s start!

Procedure:

Step1:
Place 4 resistors, 1 capacitor, 2 voltage sources, 1 npn-transistor and wire.

Step 2:
Enter the correct values for the resistors and the capacitor.
In the upper voltage source V2 choose „DC“ with an amplitude of +12V.
The lower voltage source V1 must be programmed as a sine wave with Vpeak = 100mV and 1kHz.

Step 3:
Simulate from 0 to 10ms.

Step 4:
Start the search engine and look for **BC547B.lib**“ on the internet. The content of the found html-file must be marked, copied to the clipboard and inserted into a new sheet of a text editor. Save the file as „**BC547B.lib**“ in the folder

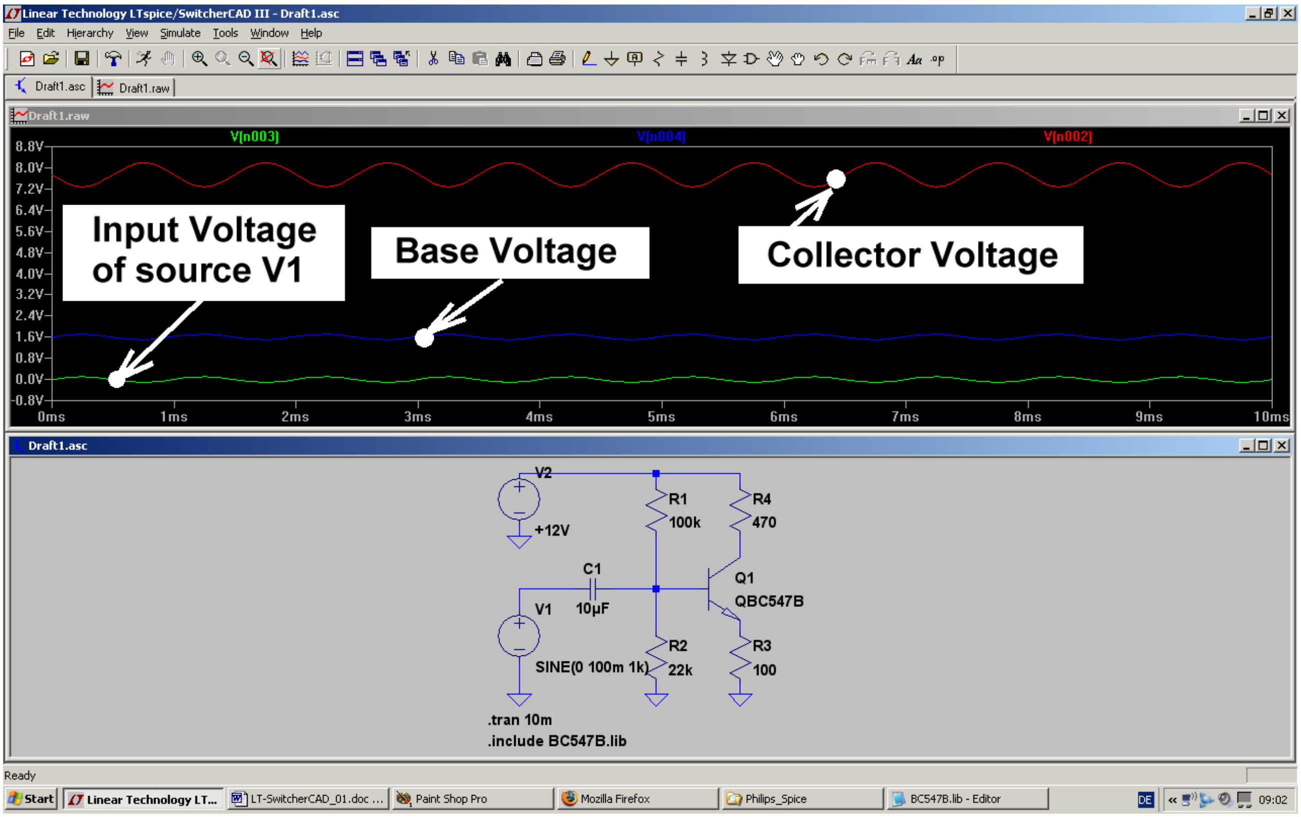
LTC / LTspiceIV / lib / sub

Step 5:
Move the cursor to the information „**NPN**“ beside the transistor’s symbol, click right and enter the correct information **„QBC547B**“ (...because this is the same as the name of the Transistor’s model file in “BC547.lib”) and add the spice directive **.include BC547.B** to your schematic.

In the simulation we would like to see the input voltage, the base and the collector voltage of the transistor. Please verify the phase shift of 180 degrees caused by the transistor.
Also we can test the " predicted gain" for such a stage, given by the simple equation

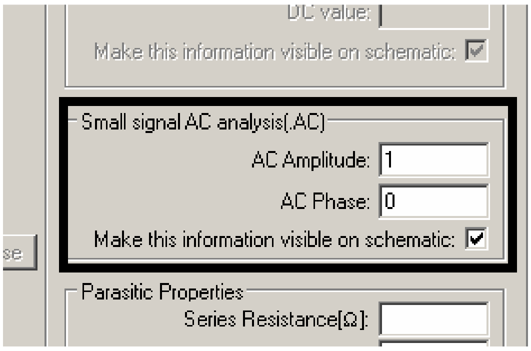
VU = RC / RE = 470 / 100 = 4,7

(to check this please measure the amplitude of the output voltage and calculate the ratio with the input voltage's amplitude of 100mV).



```
.tran 10m
.include BC547B.lib
```

9.1.2. Frequency Domain Simulation (“AC Sweep”)



DC value:

Make this information visible on schematic: ☑

Small signal AC analysis(.AC)

AC Amplitude: 1

AC Phase: 0

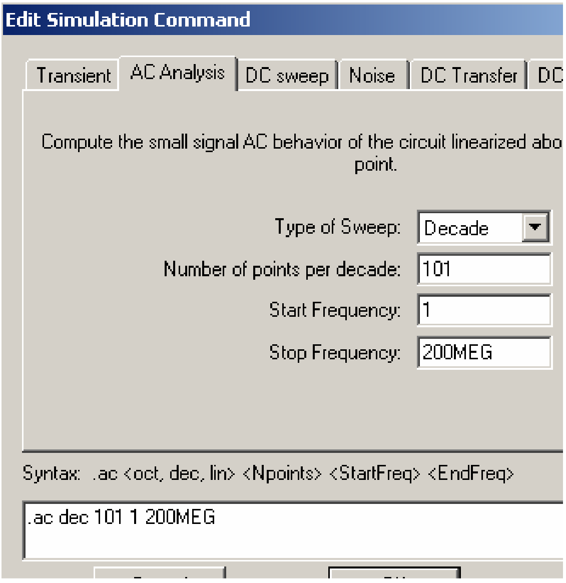
Make this information visible on schematic: ☑

Parasitic Properties

Series Resistance[Ω]:

Step 1:
Change the properties of voltage source V1 to “**AC-Sweep**” by entering the values **“1” for the AC Amplitude and “0” for the AC Phase.**

Make the information visible on the schematic.



Edit Simulation Command

Compute the small signal AC behavior of the circuit linearized abo point.

Type of Sweep: Decade

Number of points per decade: 101

Start Frequency: 1

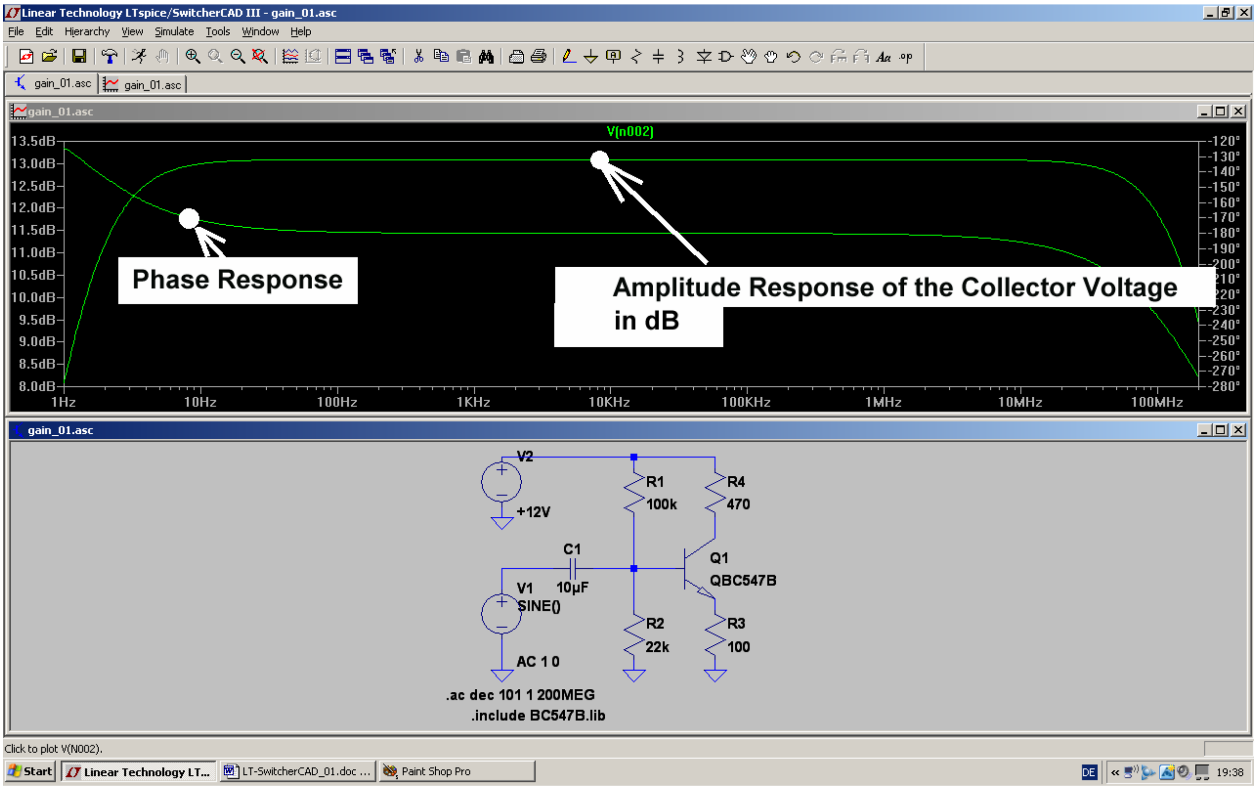
Stop Frequency: 200MEG

Syntax: .ac <oct, dec, lin> <Npoints> <StartFreq> <EndFreq>

```
.ac dec 101 1 200MEG
```

Step 2:
Open „**Simulate**“ and „**Edit Simulation Command**“ to get this menu. In „**AC Analysis**“ please enter the values for a decade sweep with 101 points per decade from 1 Hz to 200MHz.

So the simulation result should look like this:



```
.ac dec 101 1 200MEG
.include BC547B.lib
```

9.2. Two Stage Broadband Amplifier with Feedback

9.2.1. Task

Design a gainblock for a 75Ω communication system with the following specifications:

a) Input and output resistance are both 75Ω. So a lot of 75Ω blocks can be connected in series using the RG59 cable type.

b) The gain should be 6dB when loading the output by 75Ω.

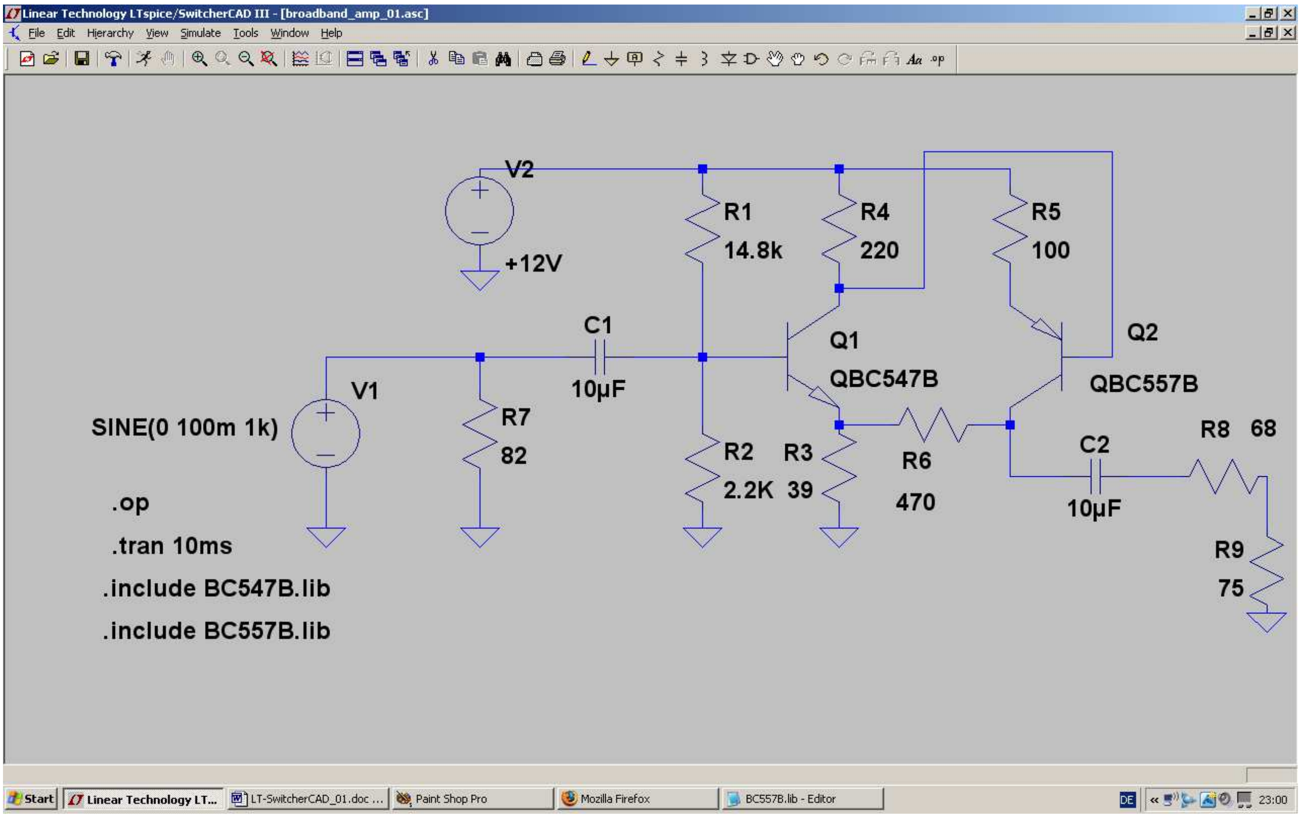
c) Use a power supply with +12V DC.

d) The gain must be constant in the frequency range from 1kHz to 10MHz.

e) Use an npn and a pnp transistor (BC547B and BC557B) with feedback.

f) Use a trimmer resistor in the base circuit of the first transistor to compensate all part tolerances.

9.2.2. Circuit Diagram for Simulation



```
SINE(0 100m 1k)

.op
.tran 10ms
.include BC547B.lib
.include BC557B.lib
```

Draw this circuit and take note:
The values of the resistors R1 and R2 must always be written as

14.8k and **2.2k**

Never write 14,8k or 2,2k -- you will get wrong results without any warnings!!

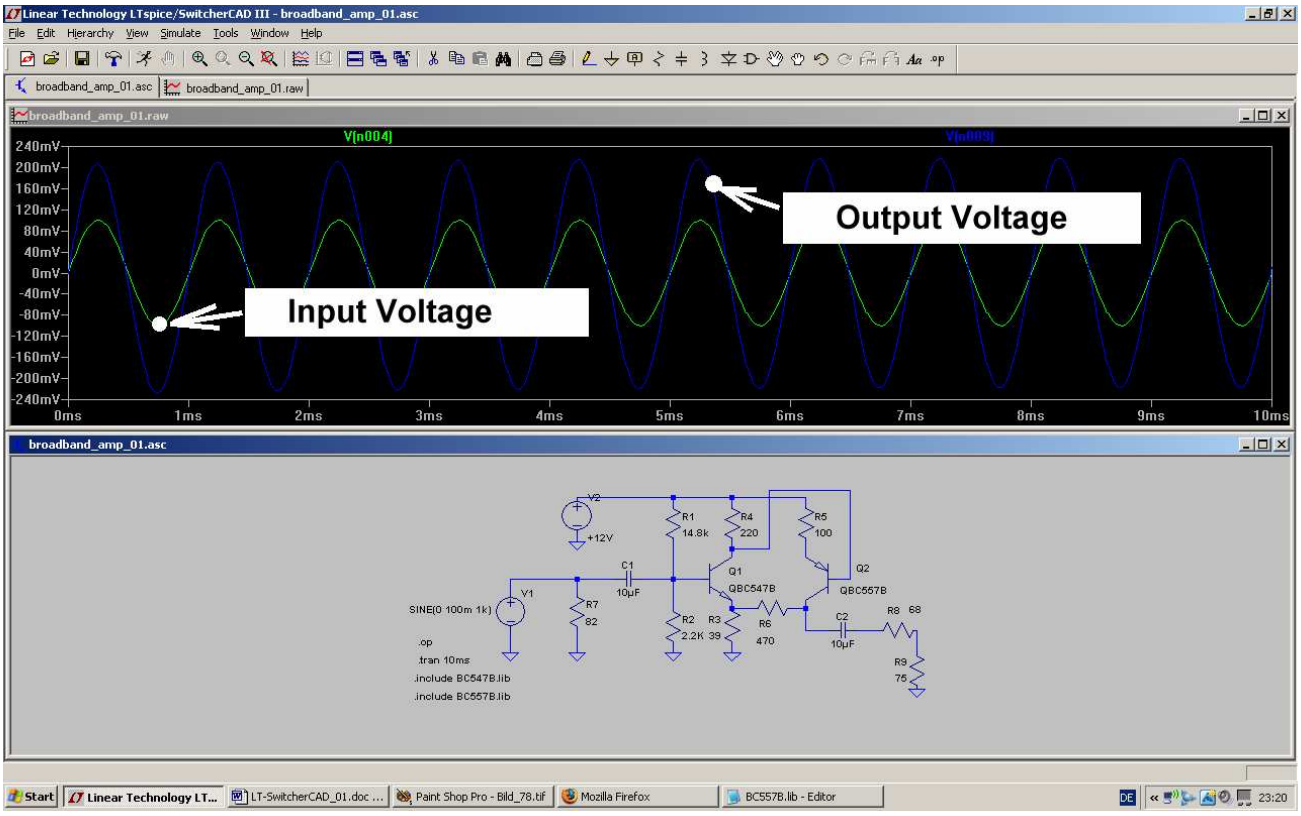
The **npn transistor BC557B** is not included in the LT's library, so you need the internet again. When youhave found the model file, copy it into an editor's new file (do you remember? You receive a HTML file) and save it in the library folder.
Note: When you have found the model files the transistors are named **QBC547B and QBC557B**. Use these names in the spice directive and in the schematic.
The input voltage is a sine wave with a peak value of 100mV and a frequency of 1kHz. Simulation time is 0….10 Milliseconds.

Something new is the command **„.op"**

With this command you get information about the DC-values in the circuit (= „DC operating point"). The information is presented in form of a list.

The test points in the schematic are the voltage source V1 and the termination resistor R9 (75Ω) at the output.

9.2.3. Time Domain Simulation

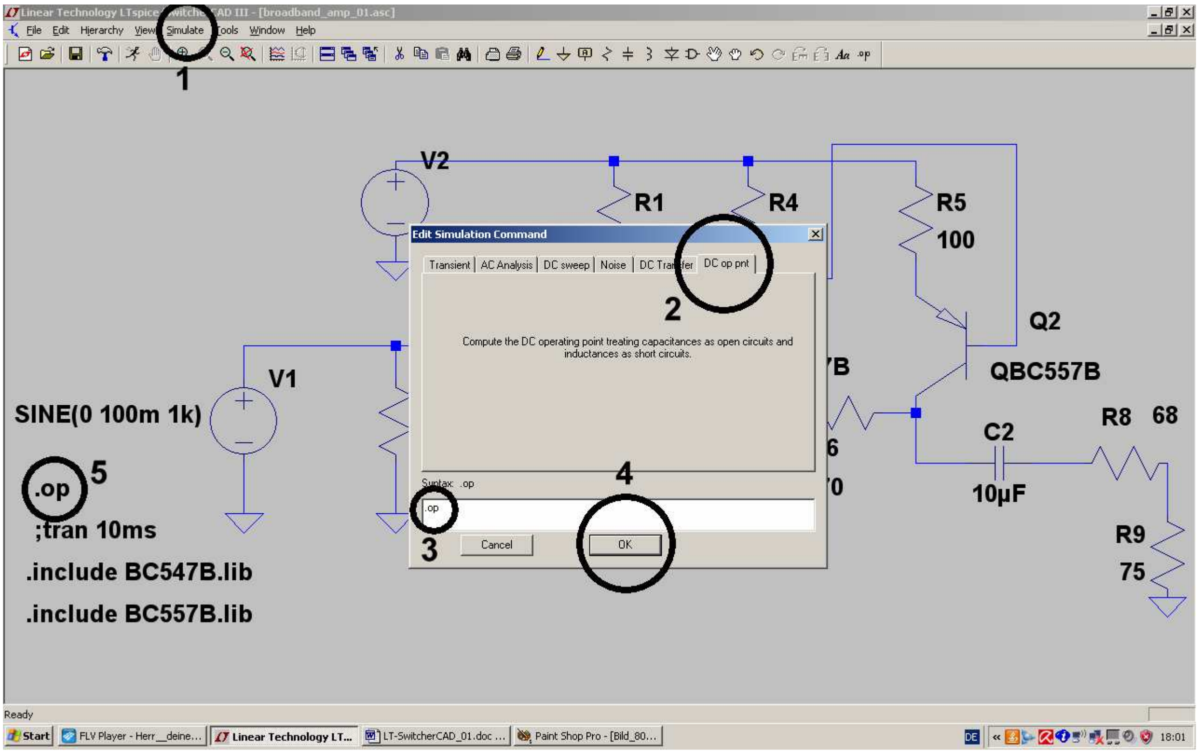


```
.op
.tran 10ms
.include BC547B.lib
.include BC557B.lib
```

As required the output voltage's peak value is twice that of the input voltage and that gives a gain of 6 dB……
Two common emitter stages produce a phase shift of 2 x 180 = 360 degrees. So the output voltage is again in phase with the input voltage.

9.2.4. DC Bias

If you want to know the DC-Currents and DC-Voltages for the operating point, then you should do this:


Linear Technology LTspice/SwitcherCAD III - [broadband_amp_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
1
V2
R1
R4
R5
100
Edit Simulation Command
Transient | AC Analysis | DC sweep | Noise | DC Transfer | DC op pnt
2
Q2
QBC557B
V1
SINE(0 100m 1k)
5
.op
;tran 10ms
.include BC547B.lib
.include BC557B.lib
C2
10µF
R8 68
R9
75
4
3
Cancel
OK
Ready
Start
FLV Player - Herr__deine...
Linear Technology LT...
LT-SwitcherCAD_01.doc ...
Paint Shop Pro - [Bild_80...
18:01

Compute the DC operating point treating capacitances as open circuits and inductances as short circuits.

Syntax: .op

.op

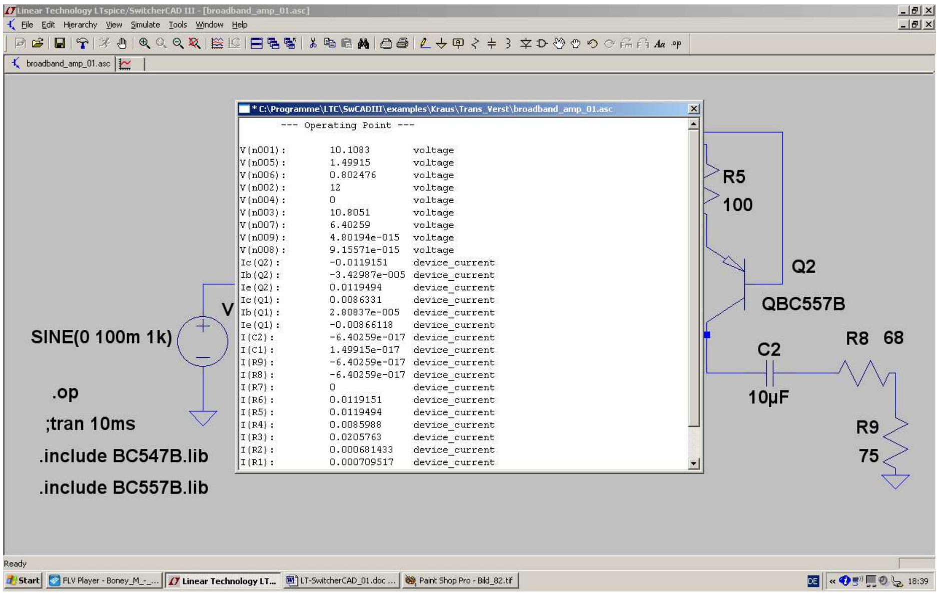
1) Open „**Simulation**“ and „**Edit Simulation Command**“.

2) Change to „**DC op point**“

3) In the window you should now find **„.op“**

4) Confirm this by pressing OK and test, whether

5) the command „.op“ is now connected to the cursor. if it is then it can be placed on the schematic. (Youcan check this as the „Tran“-Command will now start with a semicolon (**;tran 10ms**)?


Linear Technology LTspice/SwitcherCAD III - [broadband_amp_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
broadband_amp_01.asc
R5
100
Q2
QBC557B
C2
10µF
R8 68
R9
75
V
SINE(0 100m 1k)
.op
;tran 10ms
.include BC547B.lib
.include BC557B.lib
Ready
Start
FLV Player - Boney_M_-_...
Linear Technology LT...
LT-SwitcherCAD_01.doc ...
Paint Shop Pro - Bild_82.tif
18:39

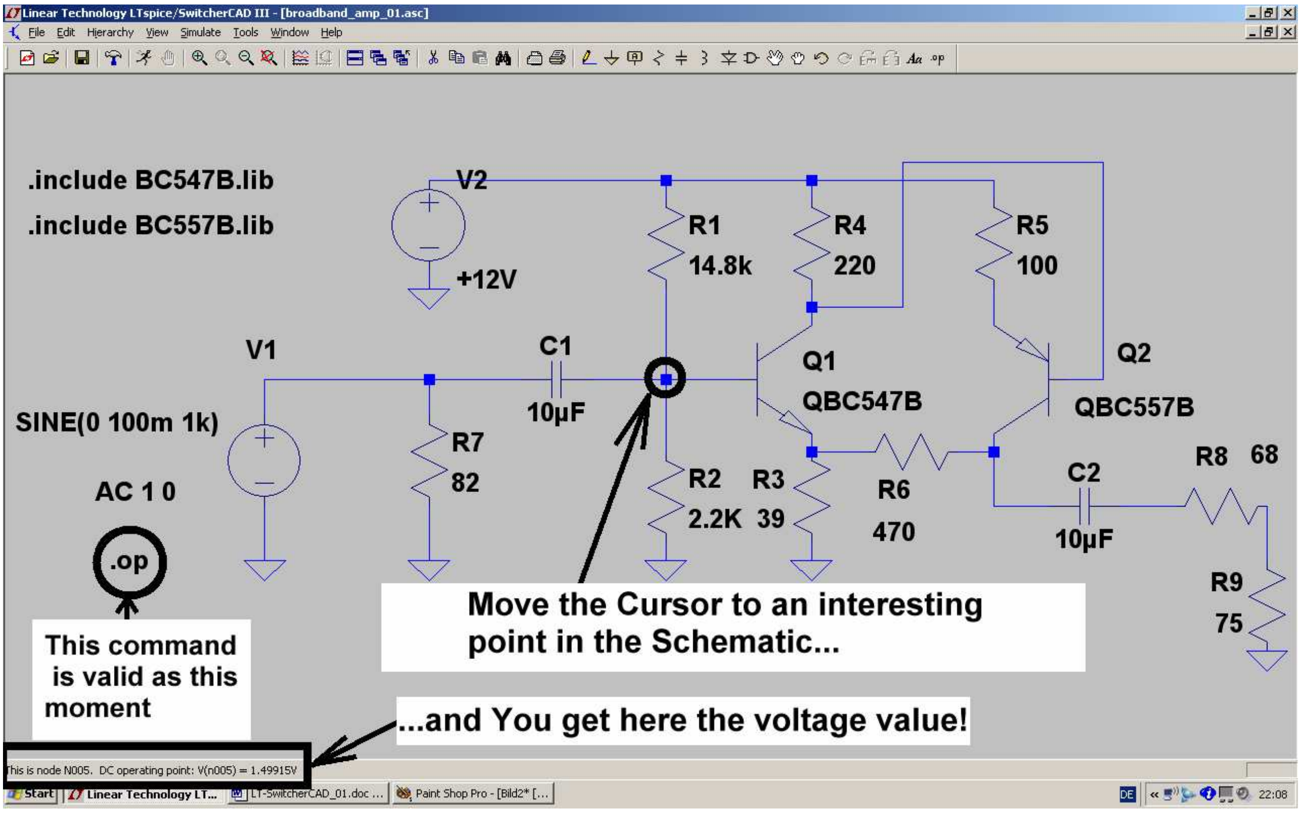
* C:\Programme\LTC\SwCADIII\examples\Kraus\Trans_Verst\broadband_amp_01.asc

```
       --- Operating Point ---

V(n001):         10.1083        voltage
V(n005):         1.49915        voltage
V(n006):         0.802476       voltage
V(n002):         12             voltage
V(n004):         0              voltage
V(n003):         10.8051        voltage
V(n007):         6.40259        voltage
V(n009):         4.80194e-015   voltage
V(n008):         9.15571e-015   voltage
Ic(Q2):          -0.0119151     device_current
Ib(Q2):          -3.42987e-005  device_current
Ie(Q2):          0.0119494      device_current
Ic(Q1):          0.0086331      device_current
Ib(Q1):          2.80837e-005   device_current
Ie(Q1):          -0.00866118    device_current
I(C2):           -6.40259e-017  device_current
I(C1):           1.49915e-017   device_current
I(R9):           -6.40259e-017  device_current
I(R8):           -6.40259e-017  device_current
I(R7):           0              device_current
I(R6):           0.0119151      device_current
I(R5):           0.0119494      device_current
I(R4):           0.0085988      device_current
I(R3):           0.0205763      device_current
I(R2):           0.000681433    device_current
I(R1):           0.000709517    device_current
```

Now press the simulation button (with the running guy) and wait for the result.
This is what it looks like and you can examine the current / voltage list.

Now close the list because there is an easier way to get every current- or voltage value (See next page):



This command is valid as this moment

Move the Cursor to an interesting point in the Schematic...

...and You get here the voltage value!

```
This is node N005.  DC operating point: V(n005) = 1.49915V
```

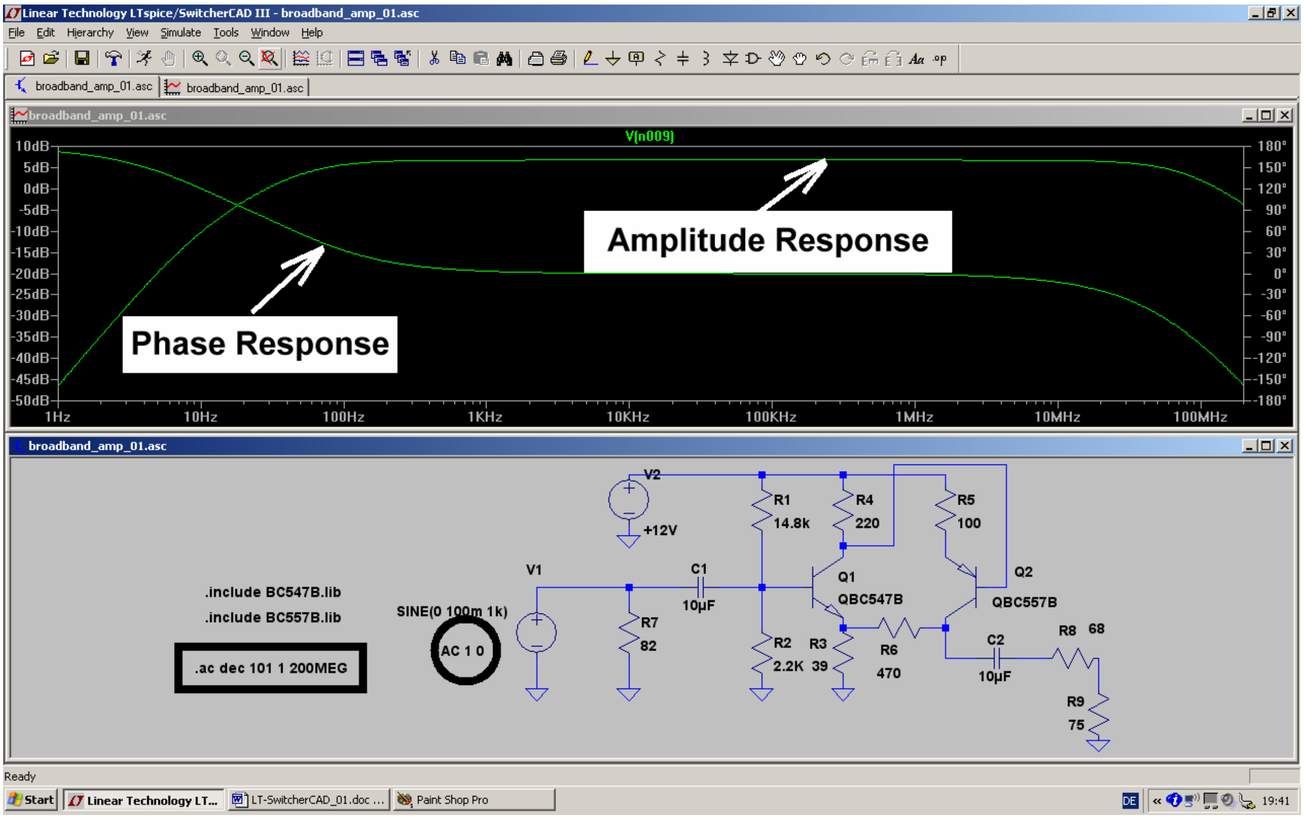
Note:

To return to the transient simulation, please replace the point in the **.op** simulation command by a semicolon. Also replace the semicolon in the **;tran** command by a point.

But the fastest way is using the path „**Simulation / Edit Simulation Command / Transient**”. There click OK and this operation will be automatically done by the software.

9.2.5. Frequency Domain Simulation: AC Sweep

No problem:
Open the property menu of the voltage source V1, enter „1“ for the AC-amplitude, „0“ for the AC-phase and a new simulation command for an AC-sweep from 1Hz to 200MHz with 101 points per decade:



.include BC547B.lib

.include BC557B.lib

.ac dec 101 1 200MEG

SINE(0 100m 1k)

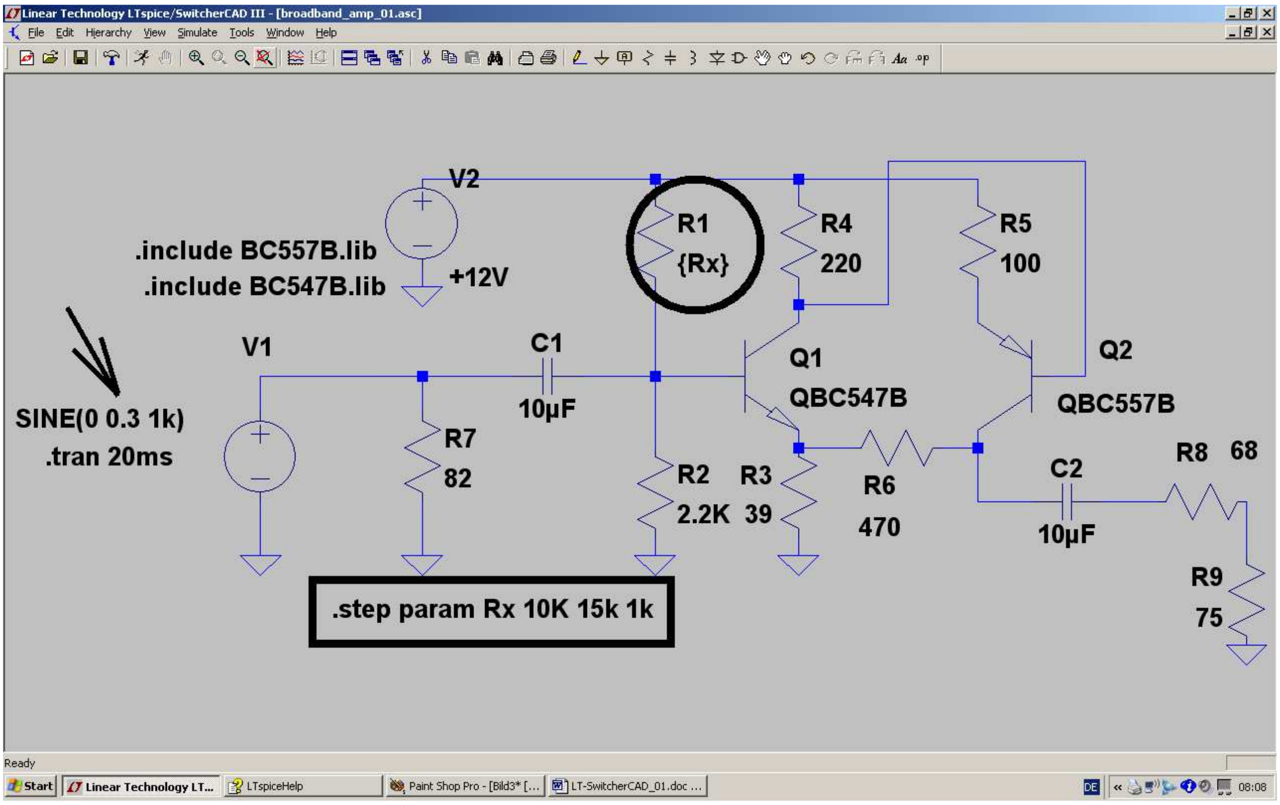
AC 1 0

9.3. Parametric Sweep

If you have a look at “**LTspice / Dot Commands**“ in the online help, then you will find a lot of commands and capabilities.

Let us test the SPICE-command **.step**

This is important because with „step“ you can automatically vary the value of a part or a voltage and repeat the simulation.
Let’s take the circuit of the last section and vary the upper resistor R1 in the base voltage divider in 5 steps from 10k to 15k. This will enable you to get the optimum operating point.



```
.include BC557B.lib
.include BC547B.lib
SINE(0 0.3 1k)
.tran 20ms
.step param Rx 10K 15k 1k
```

Linear Technology LTspice/SwitcherCAD III - [broadband_amp_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
V2
+12V
R1
{Rx}
R4
220
R5
100
V1
C1
10µF
Q1
QBC547B
Q2
QBC557B
R7
82
R2
2.2K
R3
39
R6
470
C2
10µF
R8 68
R9
75
Ready
Start
Linear Technology LT...
LTspiceHelp
Paint Shop Pro - [Bild3* [...
LT-SwitcherCAD_01.doc ...
DE
08:08

Note:

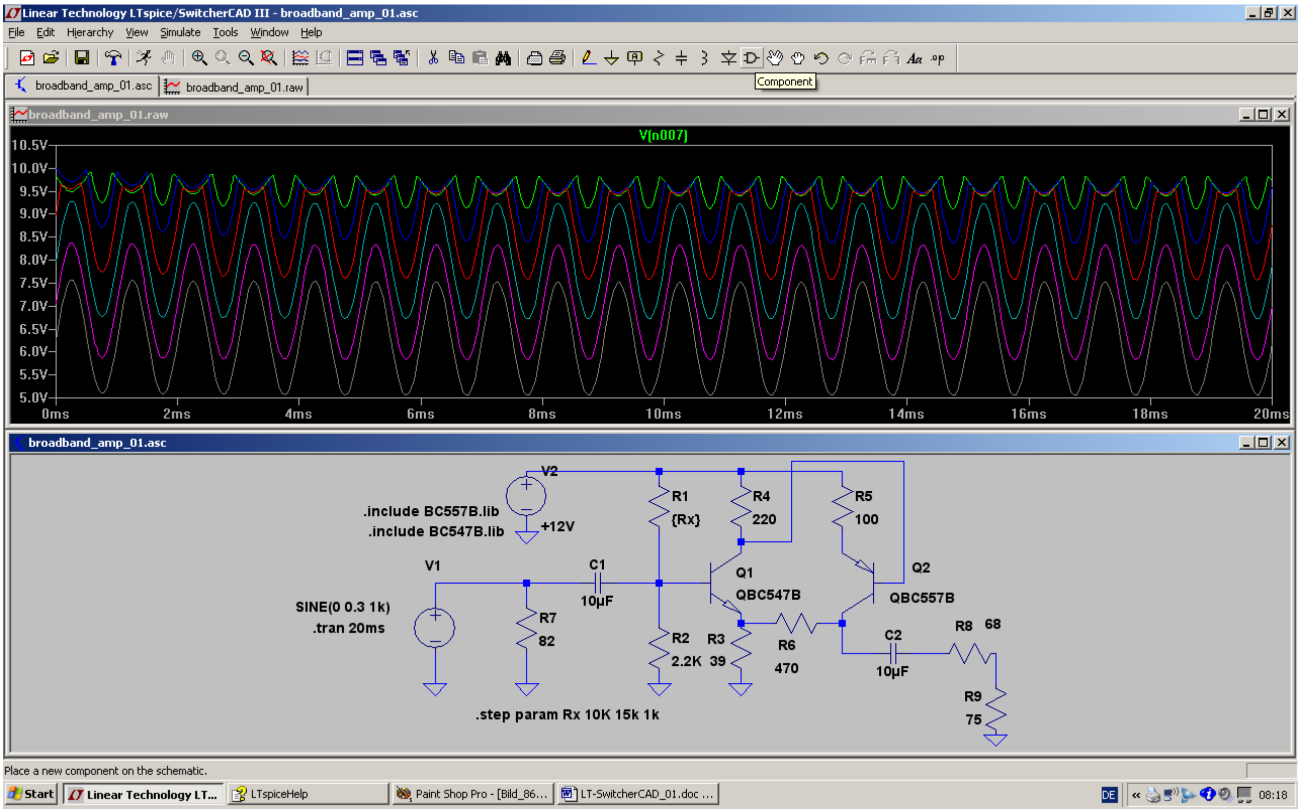
a) The peak value of the input voltage is now 0.3V

b) The value of R1 is replaced by the variable **{Rx}**

c) With the spice directive **.step param Rx 10k 15k 1k** the value of Rx is increased from 10k to 15k in steps of 1k.

Now press the simulation button (the “running guy”). The simulation will run, but you don’t see any results. So move your cursor to the collector of the transistor Q2 and click.

You will see this result and that a value of approx. 15k for R1 (= lowest curve) will do the job perfectly (and give a V_{DC} = +6.5V at the collector of Q2)



```
.include BC557B.lib
.include BC547B.lib
.tran 20ms
.step param Rx 10K 15k 1k
```

Place a new component on the schematic.

10. Project 6: OP Amp Circuits

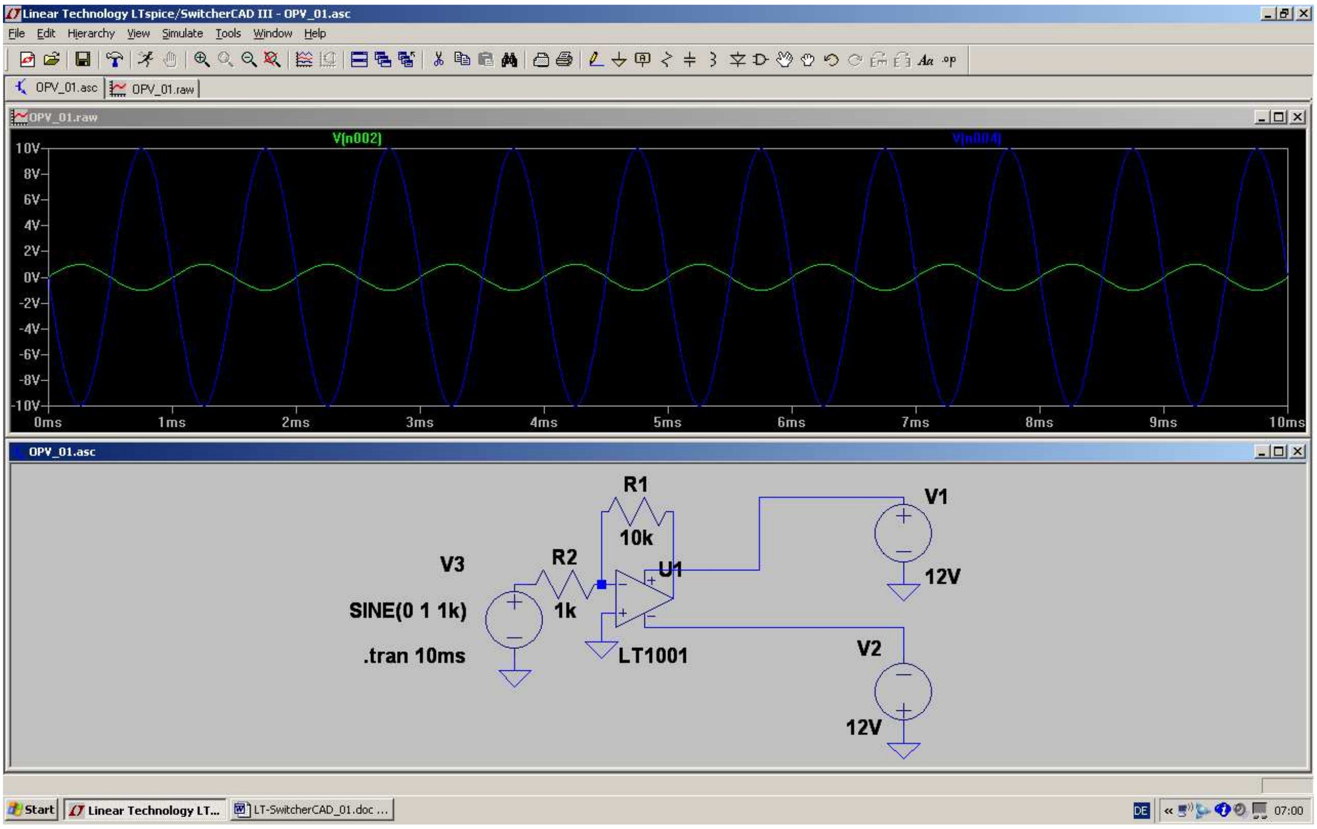
10.1. Let us start with an Inverting Amplifier

Let us have a look at a simple OP Amp circuit using one of the LT models which come with the software.

Task:
To simulate an inverting amplifier with the „LT1001“ to have the following specifications:

Gain = 10 (= 20dB)
Input Resistance = 1 kΩ
Use two power supplies with +15V and -15V
The input signal is a sine wave with a peak value of 1V and a frequency of 1kHz. Simulate the input and output voltage from 0...10ms.

Solution:



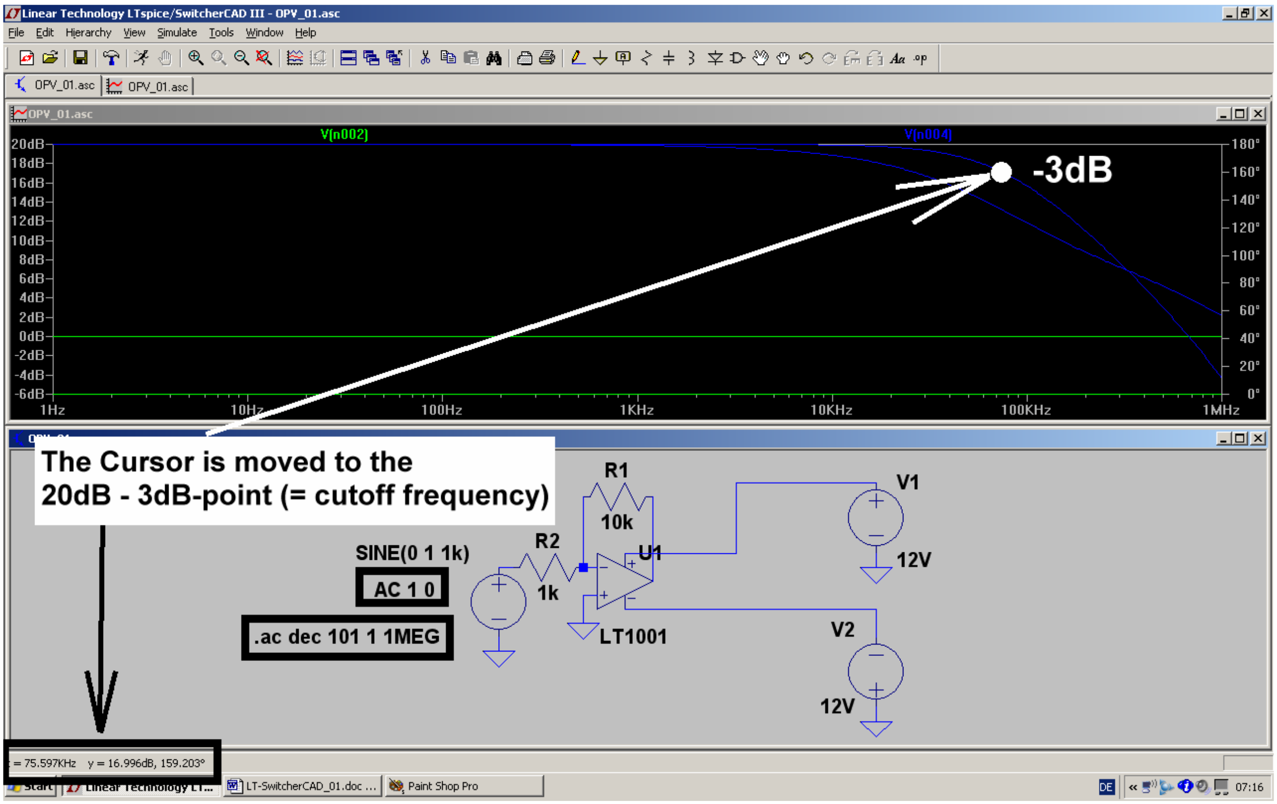
You see, as predicted by theory, that there is a phase shift of 180 degrees between input and output.

Additional task:
Use an AC sweep to determine the frequency response and the upper cutoff frequency. Simulate up to 1MHz.

Solution:
In the properties of the voltage source V3 set **AC amplitude = 1 and AC phase = 0**
.
Then write the simulation command

```
.ac dec 101 1 1MEG
```

for a decadic sweep with 101 points per decade from 1 Hz to 1MHz.



The Cursor is moved to the 20dB - 3dB-point (= cutoff frequency)

```
.ac dec 101 1 1MEG
```

```
= 75.597KHz   y = 16.996dB, 159.203°
```

Using the cursor you will find the corner (= cutoff) frequency at 75.6kHz. Here the gain has decreased by -3dB.

10.2. Preparing a SPICE Model from the Internet

When doing your own development work you often need a special circuit from a manufacturer. Most companies offer their SPICE model collection free of charge on the Internet. So search for and use them!
Modern OP Amps can work up to 500MHz and more - even the circuit's noise can be simulated. Let us use such a modern part in a simulation and learn how to create a symbol for a downloaded complex circuit.

10.2.1. Gainblock for 1kHz to 30 MHz with OPA355

The OPA355 is a modern low noise OP Amp with a cutoff frequency of 200MHz (there the gain has decreased to 1). We want to create a „gainblock" for a 75Ω – system with it.

Specifiations:

a) Only one power supply with +5 V (= maximum Vcc value). Therefore the mean output DC voltage must be set to +2.5V.

b) Gain = 4 for an unloaded output, but gain = 2 with a terminated output pin.

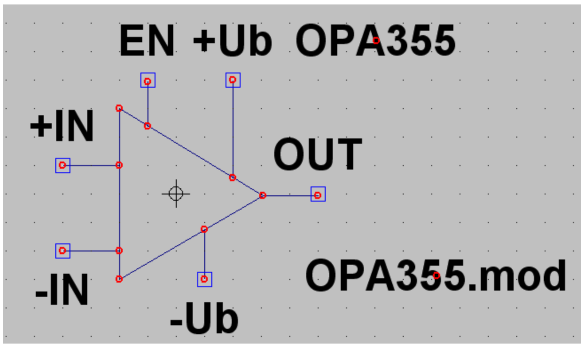
c) Input resistance = output resistance = 75Ω.

d) Lower corner frequency = 100 Hz. Upper corner frequency = 50 MHz for a gain reduction of 0.5dB.

e) Use a non inverting circuit.

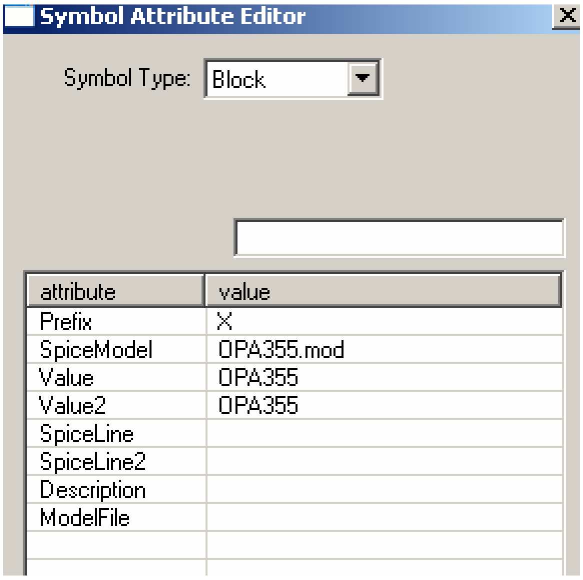
10.2.2. Simulation using the selfmade OPA355 Subcircuit Model

The necessary SPICE model can be found as „sbom143.zip" in the Texas Instruments homepage. After de-zipping save it as „**OPA355.mod**" in the well-known library folder „**lib / sub**".



Once more you have to create a new symbol (See chapter 6.2.3).
Be aware of the additional **ENABLE pin. In the schematic this pin must be connected to the supply voltage (+5V):**

This should be the result....



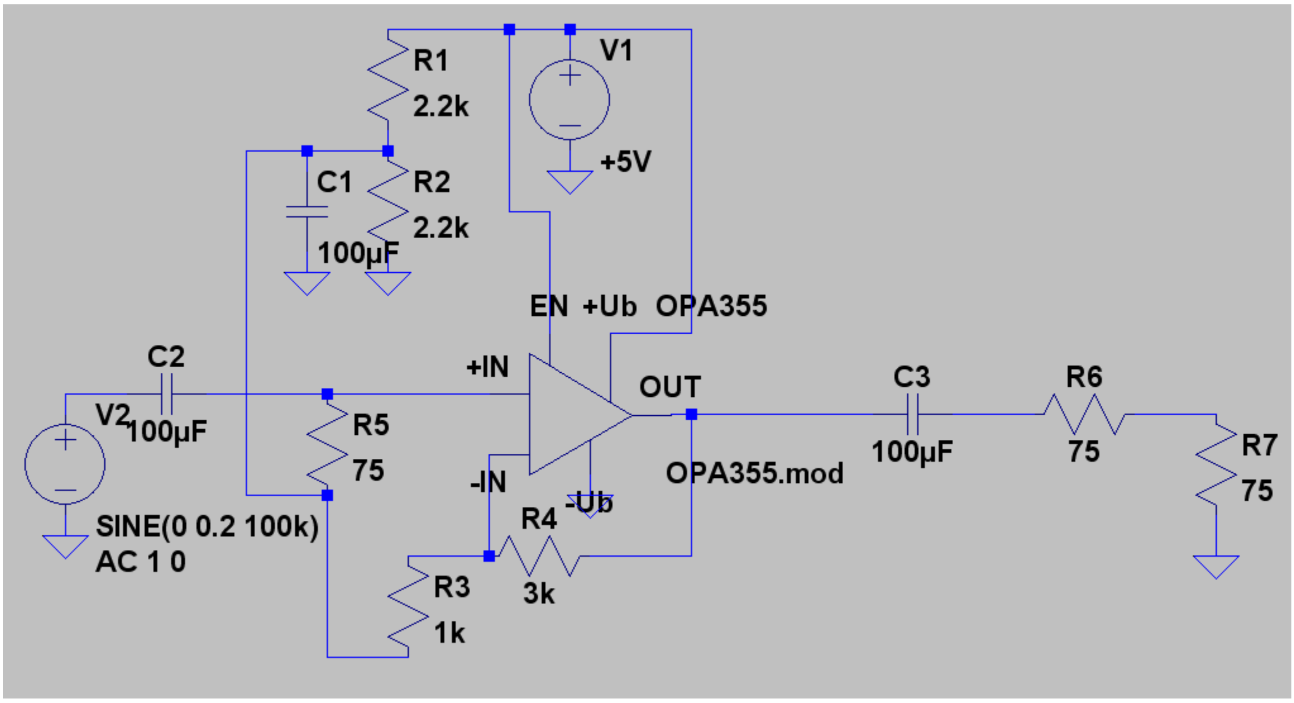
Symbol Type: Block

attribute	value
Prefix	X
SpiceModel	OPA355.mod
Value	OPA355
Value2	OPA355
SpiceLine	
SpiceLine2	
Description	
ModelFile	

…and this is the correct attribute list. Only with these entries will everything run well.

Warning:
The value for the attribute "Value2" MUST NEVER BE EMPTY!!!
(…the best is to use the SPICE model name…)
If this entry is missing, then you can get cryptic error messages and the simulation aborts…

Let us now draw the complete circuit:



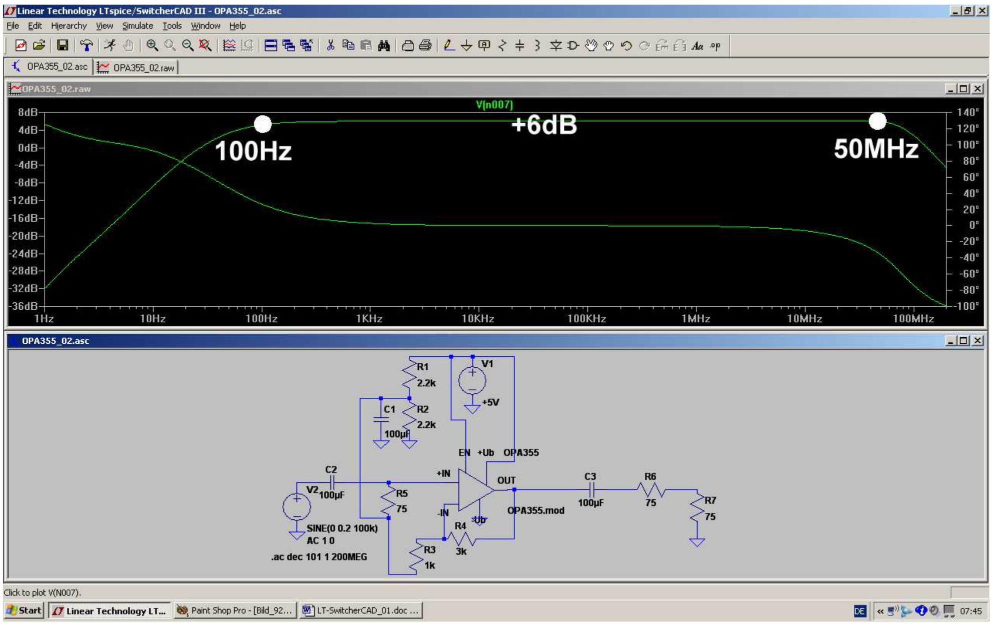
Explanations:

a) The +5V supply must be connected to the pins „+Ub“ and „Enable“. Also the voltage divider (consisting of R1 and R2) must be connected to +5V to get the necessary +2.5V at the input pins.

b) R5 gives the input resistance of 75Ω, R6 the output resistance of 75Ω.

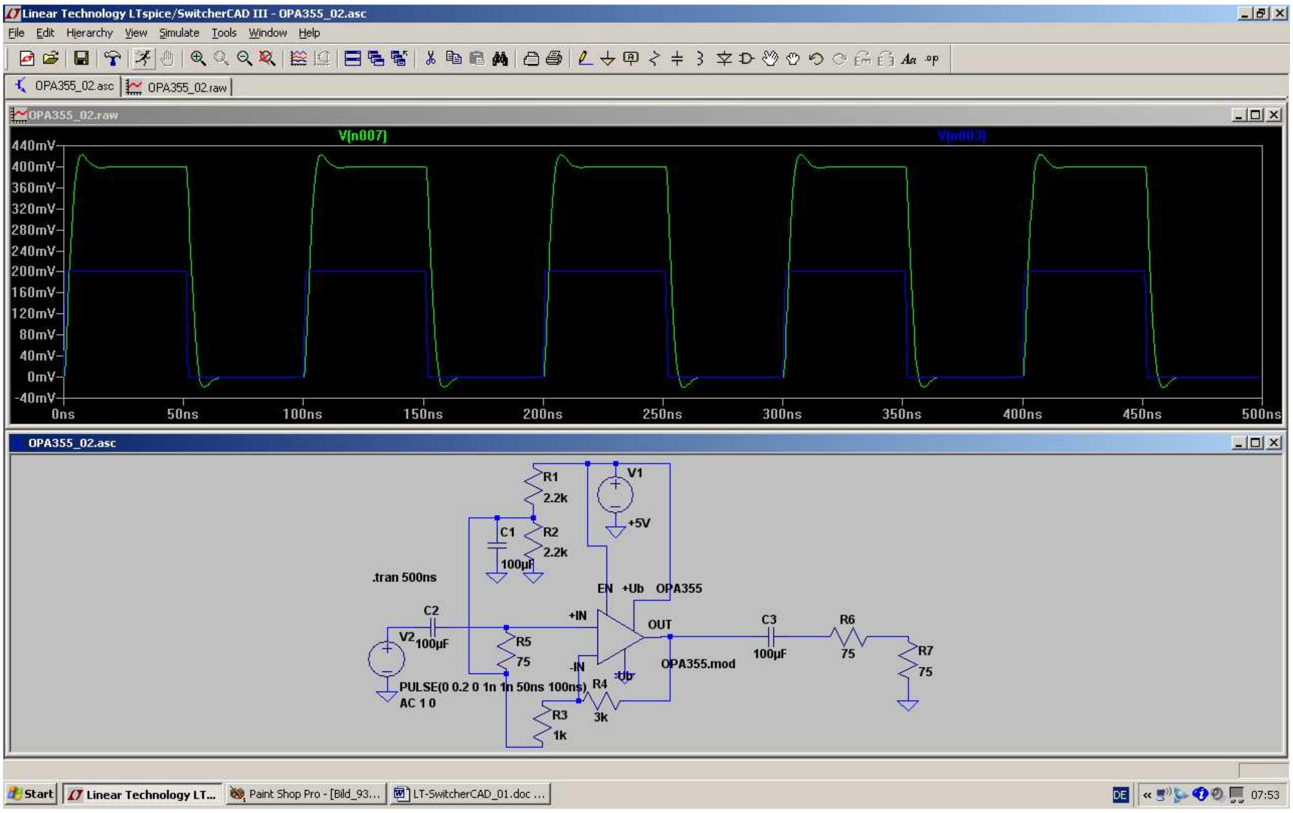
c) R4 and R3 form a feedback network. So the „non-inverting function“, and an unloaded gain =4 and a loaded gain = 2 (+6dB) are realised

This is the result of the simulation for the output voltage at R7:



Fine!

And something interesting is to feed the circuit with a pulse voltage (peak value = 0.5V / frequency = 10MHz):



.tran 500ns

PULSE(0 0.2 0 1n 1n 50ns 100ns)
AC 1 0

Very nice, indeed

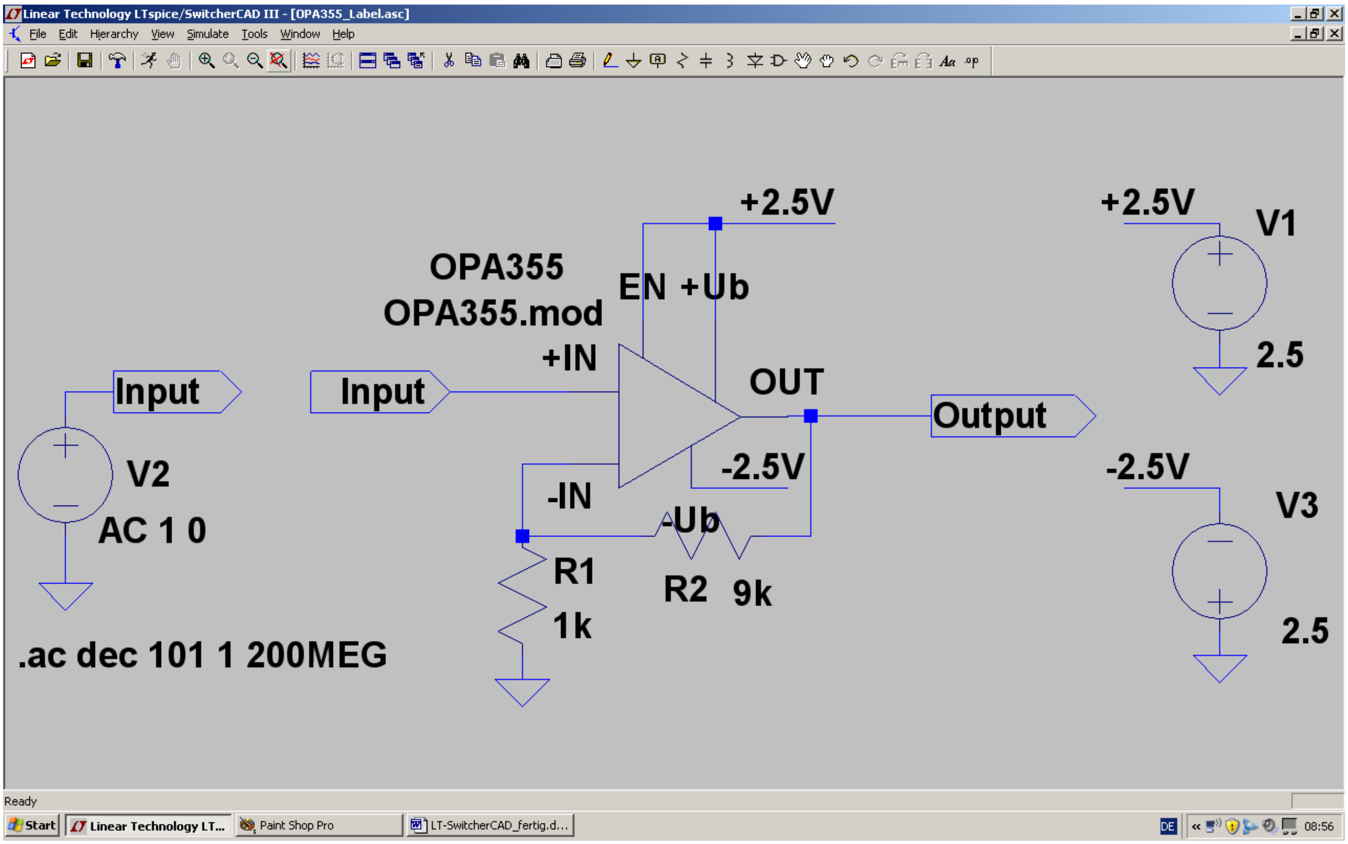
10.3. Use of Labels

In general OP Amp circuits need a positive and a negative supply voltage. The fully drawn circuit then gets a little bit untidy, so applying some ***Labels*** simplifies the appearance.
When using a label you replace the wiring by a simple connection with a name for the node at all points where it is needed and you no longer need to draw all the necessary wires.

Let us use the OPA355 from the last chapter and build a new gain block (gain = 10 (+20dB) ; non- inverting circuit; 2 supplies of +2.5V and -2.5V).
Also use a label for the input and for the output nodes.

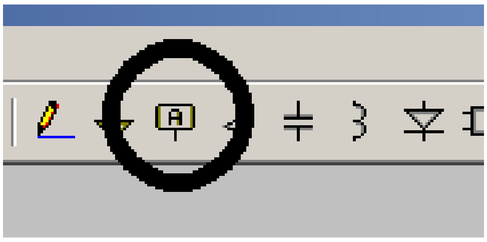
This has another advantage:
In the waveform viewer you now get the label as information for the trace instead of the node number....

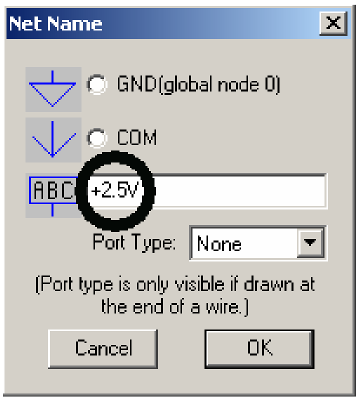
So the schematic looks like this, preset for a sweep from 1Hz to 200MHz:



And now let us do the work, starting with the 2 power supplies.

Step 1:
Delete all wiring between the power supply and the circuit and press the “label button”:





+2.5V

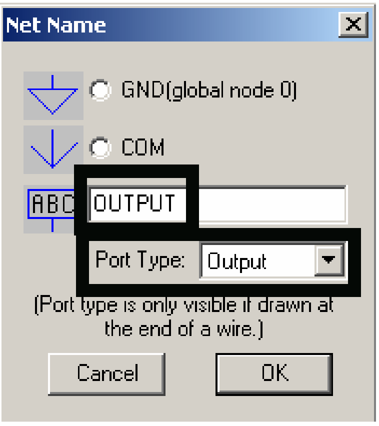
Port Type: None

Step 2:

Now enter **+2.5V** in the field and click on OK.
The label “hangs” on the cursor and can be placed on the schematic and wired.

Note:
Before placing the label, it looks like a huge rectangle with an additional little rectangle on the centre of the lower edge. This little rectangle is the “catching point” and must be connected to the symbol of the power supply!

Step 3:
Now click and place the label at every point in the schematic which needs the +2,5V supply.When all the instances are placed then connect each one to the corresponding circuit point.
Note: if wires are already present the labels can be placed directly on the wire using the “catching point” and they will automatically be connected.
Repeat steps 2 & 3 for the “-2.5V supply



OUTPUT

Port Type: Output

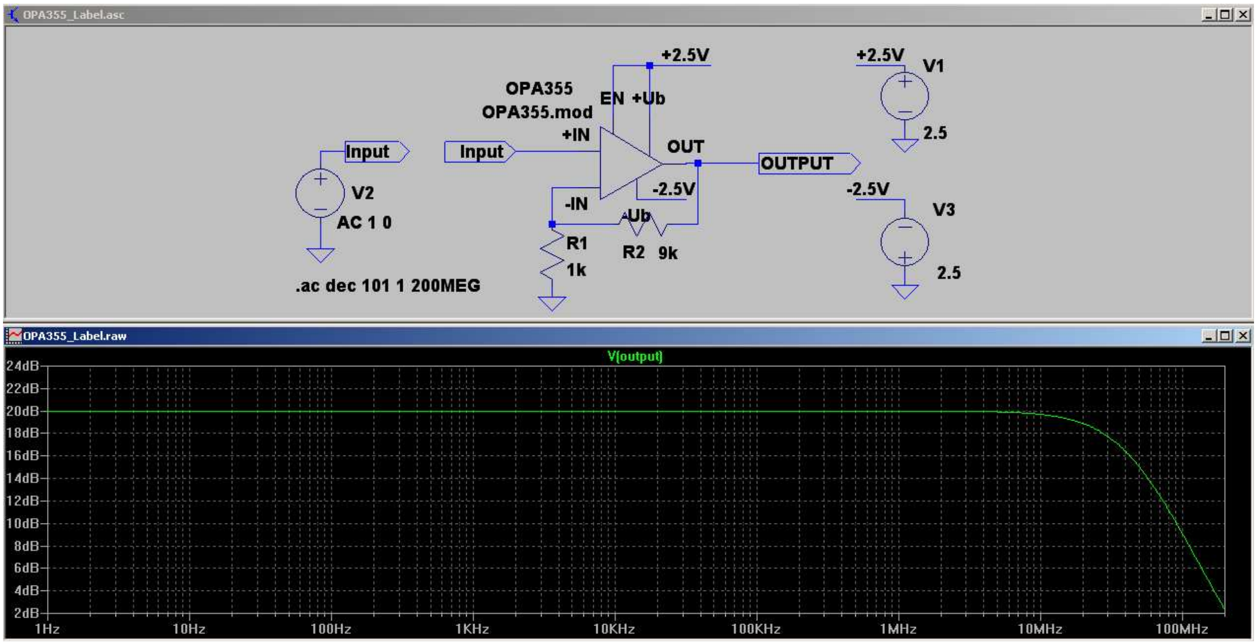
Lastly, repeat this procedure to place the „input“- and „output“ labels.

But please note:

The input of the OP Amp needs the label „INPUT“-- but select the Port Type as „Input”.

The output of the OP Amp needs the label “OUTPUT” -- but select the Port Type as “Output”.

And this is the sweep result (only gain, without phase):



11. Project 7: DC-DC Converters

First a **warning**:
These circuits use complex simulations and realistic inductance properties because an oversimplified inductor model will cause errors or simulation aborts in many applications. Very expensive simulation software versions use special (and secret) tools to avoid these problems, but we have only the normal Berkeley SPICE kernel......

With the following rules you can avoid most of these problems:

a) Never use an ideal inductance -- always add a series resistor (even 0.01Ω can do the job!).

b) Every inductor has a winding capacitance, and often the circuit has additional stray capacitance. This always gives **resonance effects** with (sometimes) unexpected results and problems during the simulation. So use damping resistors or long simulation times to get correct simulation results.

11.1. Model for the power-MOSFET „IRFZ44N“

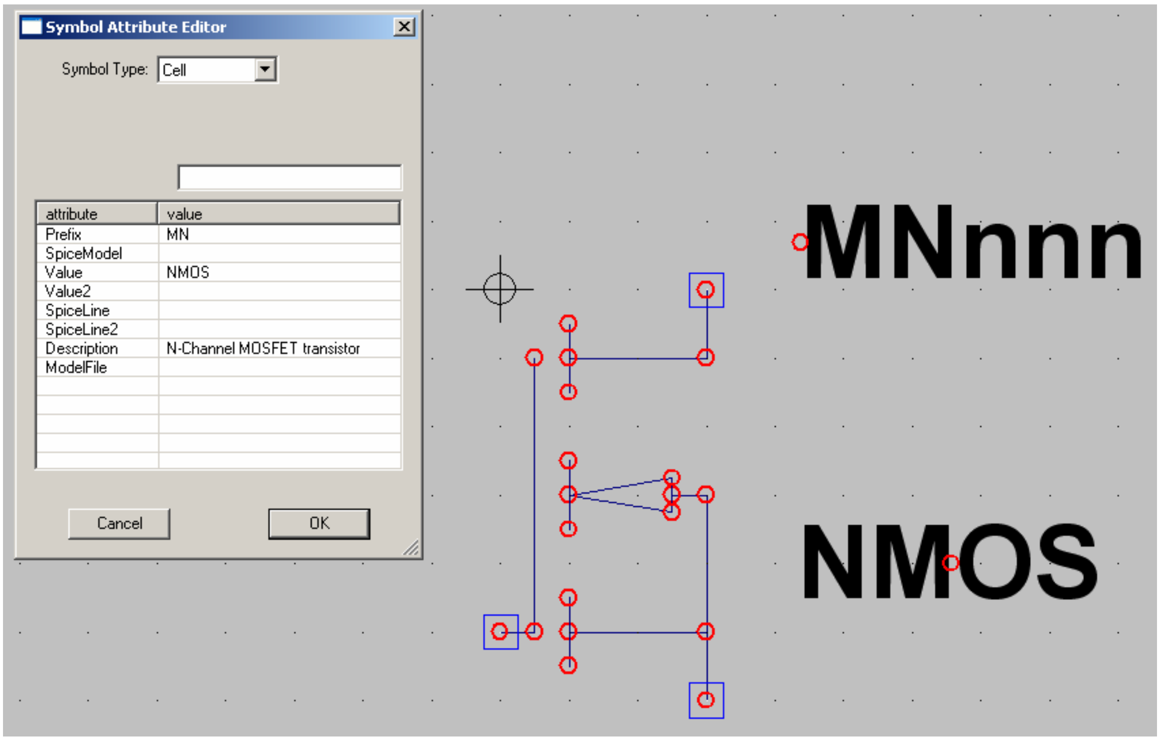
For all the following experiments we need a good and fast electronic switch -- today this is always a power MOSFET. Choose the IRFZ44N (produced by IRF = International Rectifiers) which is a good choice and widely available part.

Sorry, but now you have to create a symbol again

Let’s go:

Step 1:
Search with Google for “**irfz44n spice model**“ and find the file “**irfz44n.spi**“. Mark the content, copy into a new editor file and save this file as “**irfz44n.lib**“ in the folder “**lib / sub**“ of LTspice.

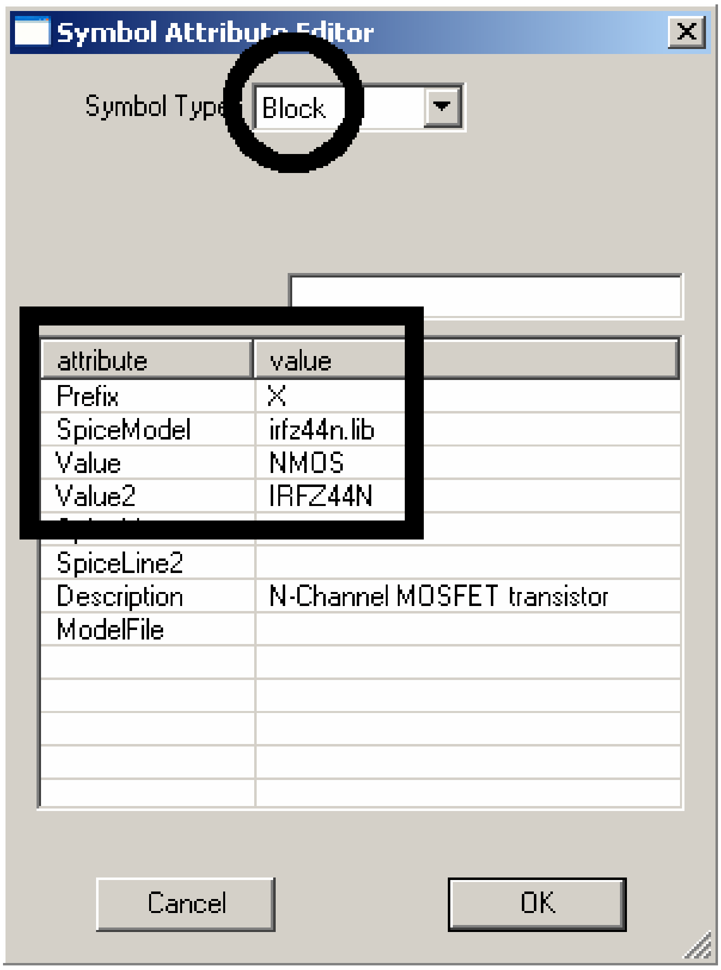
Step 2:
Choose “**New Symbol**“ in the “**File**“Menu. Open “**File**” once more and -- with “open” -- follow the path “**lib / sym**“ and find “**nmos.asy**“ (.To see it you must make “All Files” visible). Then open the attribute list of this part by following the path **“Edit / Attribute / Edit Attributes**“.



Symbol Attribute Editor

Symbol Type: Cell

attribute	value
Prefix	MN
SpiceModel	
Value	NMOS
Value2	
SpiceLine	
SpiceLine2	
Description	N-Channel MOSFET transistor
ModelFile	



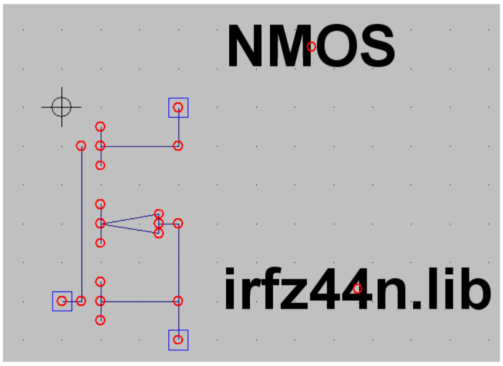
Symbol Attribute Editor

Symbol Type: Block

attribute	value
Prefix	X
SpiceModel	irfz44n.lib
Value	NMOS
Value2	IRFZ44N
SpiceLine	
SpiceLine2	
Description	N-Channel MOSFET transistor
ModelFile	

Step 3:
And now carefully modify the entries as given in the figure on the left.

(Pay special attention to the first line: „Prefix“ must be changed from MN = model to **„X“ = subcircuit)**

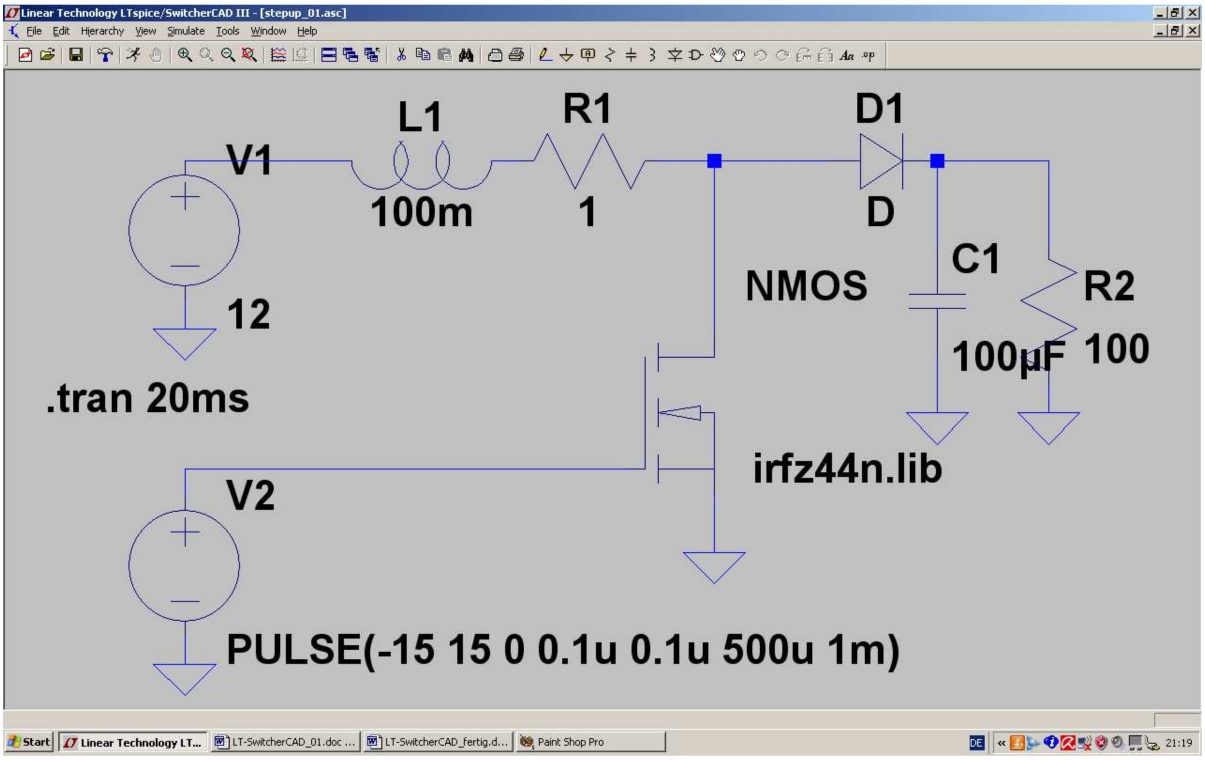


Step 4:
Open „**Edit**“ and „**Attribute Window**“. Click on „**Value**“ and place the description „**NMOS**“ over the symbol.
Then press OK, repeat the procedure but click now on „**SPICE File**“. The information „**irfz44n.lib**“ must now be placed beside or below the Symbol.

Now save the complete symbol -- It is best to create a new folder „**NMOS-FETs**“ in „**lib / sym**“ and to save the new symbol in it.

That’s all -- now let’s tackle the first task..

11.2. The Step Up Converter


Linear Technology LTspice/SwitcherCAD III - [stepup_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
L1
R1
D1
V1
100m
1
D
C1
NMOS
R2
12
100µF
100
irfz44n.lib
V2
Start
Linear Technology LT...
LT-SwitcherCAD_01.doc ...
LT-SwitcherCAD_fertig.d...
Paint Shop Pro
DE
21:19

.tran 20ms

PULSE(-15 15 0 0.1u 0.1u 500u 1m)

The schematic shows a +12V power supply , followed by an inductance with L = 100mH and the obligatory series resistor of 1Ω.
The Power-MOSFET acts as an electronic switch between R1 and D1 to connect this node to ground for a definite time.

When switched ON, current flows from the power supply through the inductor and its series resistor to ground and rises linearly (...this is true, when the voltage across an inductor is constant) .
So more and more magnetic energy is stored in the inductor and this can be calculated from:

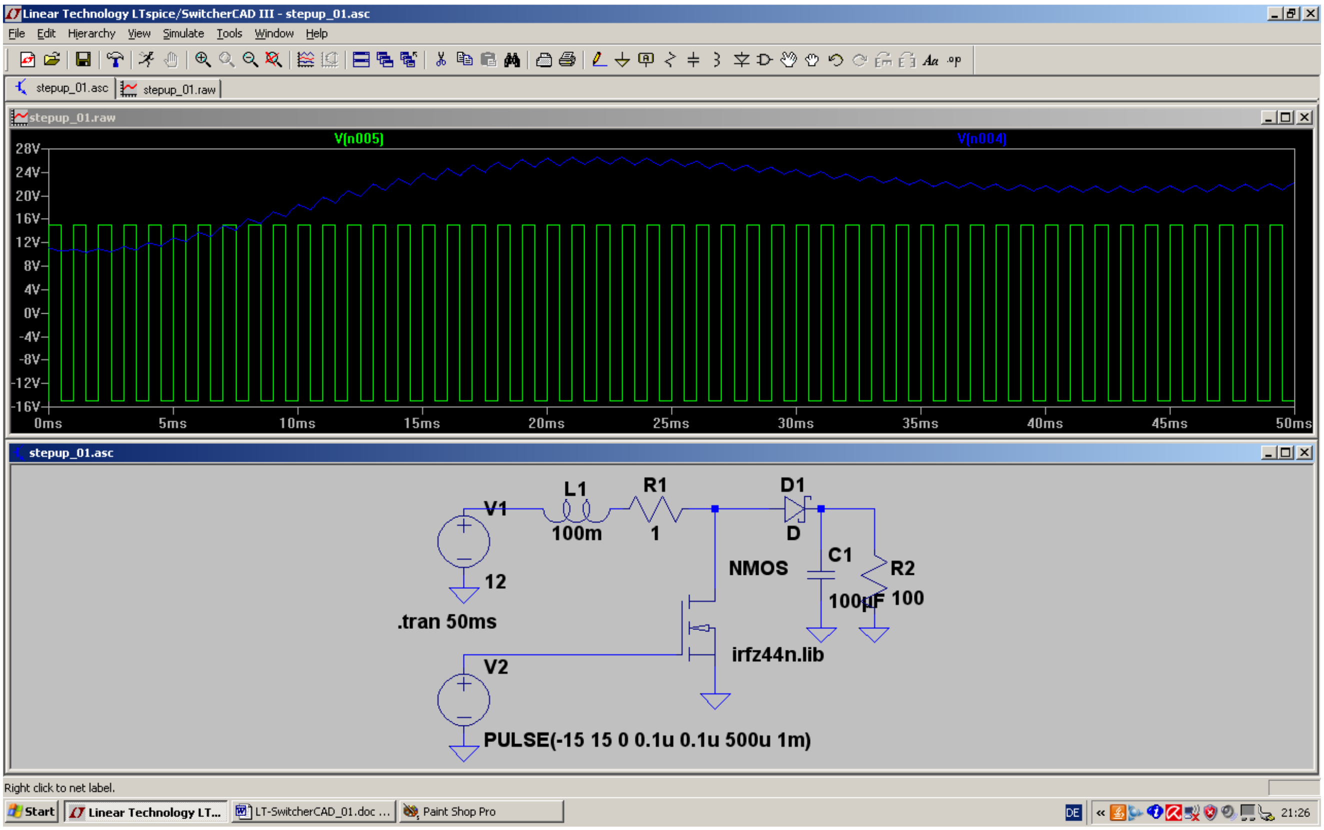
$$E = \frac{1}{2} \bullet L \bullet I^2$$

When the FET is switched OFF, this energy cannot dissipate suddenly and the inductor’s current cannot switch rapidly (….this would cause a nearly infinite induced voltage...) and so stays at the same value.
That means that the inductor current now flows to the right hand side, switches ON the schottky diode, charges the output capacitor and flows through the output resistor R2. When the gate of the MOSFET is fed with a square wave voltage, the output voltage rises until a “balance” is reached between picked-up-energy (by the Inductor) and the energy delivered to the output resistor R2.

Important: the amplitude of the output voltage is the sum of supply voltage +12V and the inductors voltage!

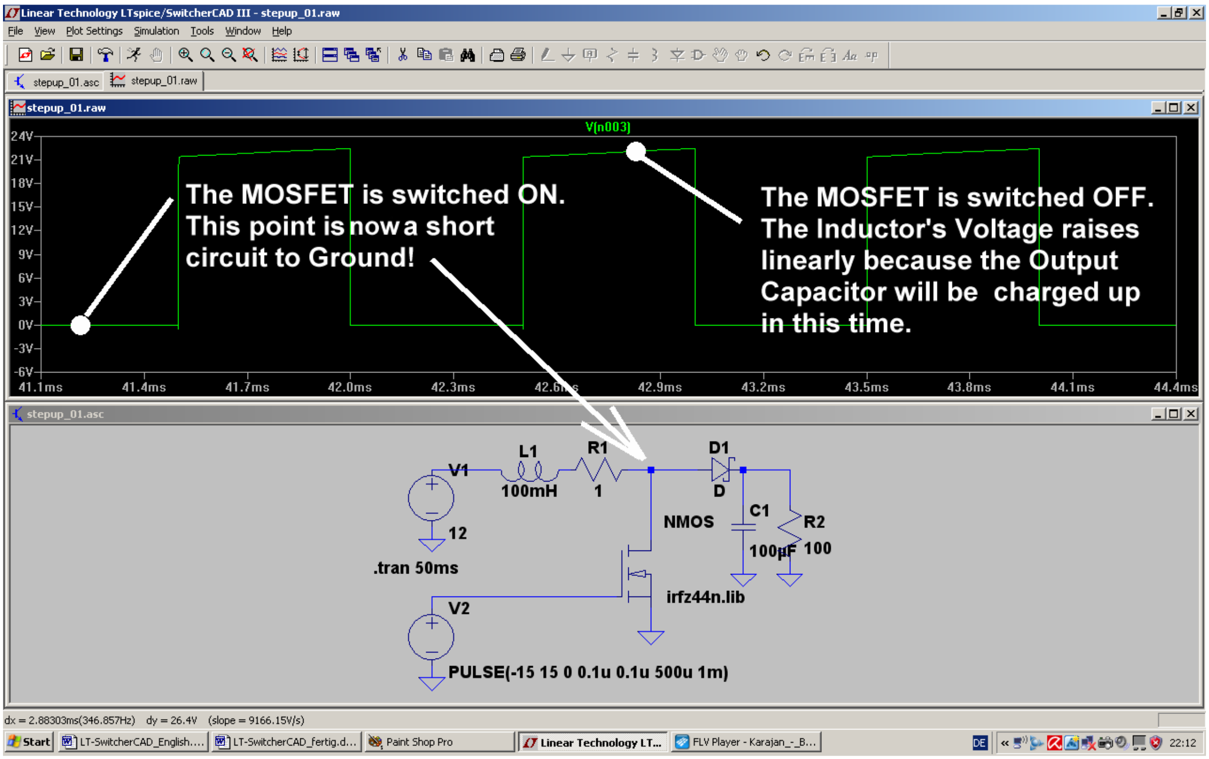
This means that the output voltage of this circuit will always be higher than the supply voltage!

Now draw the schematic and feed the gate with a symmetric square wave voltage (frequency = 1 kHz / (Vmax) = +15V / Vmin = -15V). Choose a simulation time from 0.....50ms. Show the gate voltage and the output voltage.



Clearly, it can now be seen that the output voltage is a mixture of a mean DC Value of +23V and a small „sawtooth-ripple“ (= charging and discharging of the output capacitor).

Lastly, let us have a look at the drain voltage of the MOSFET:



The MOSFET is switched ON. This point is now a short circuit to Ground!

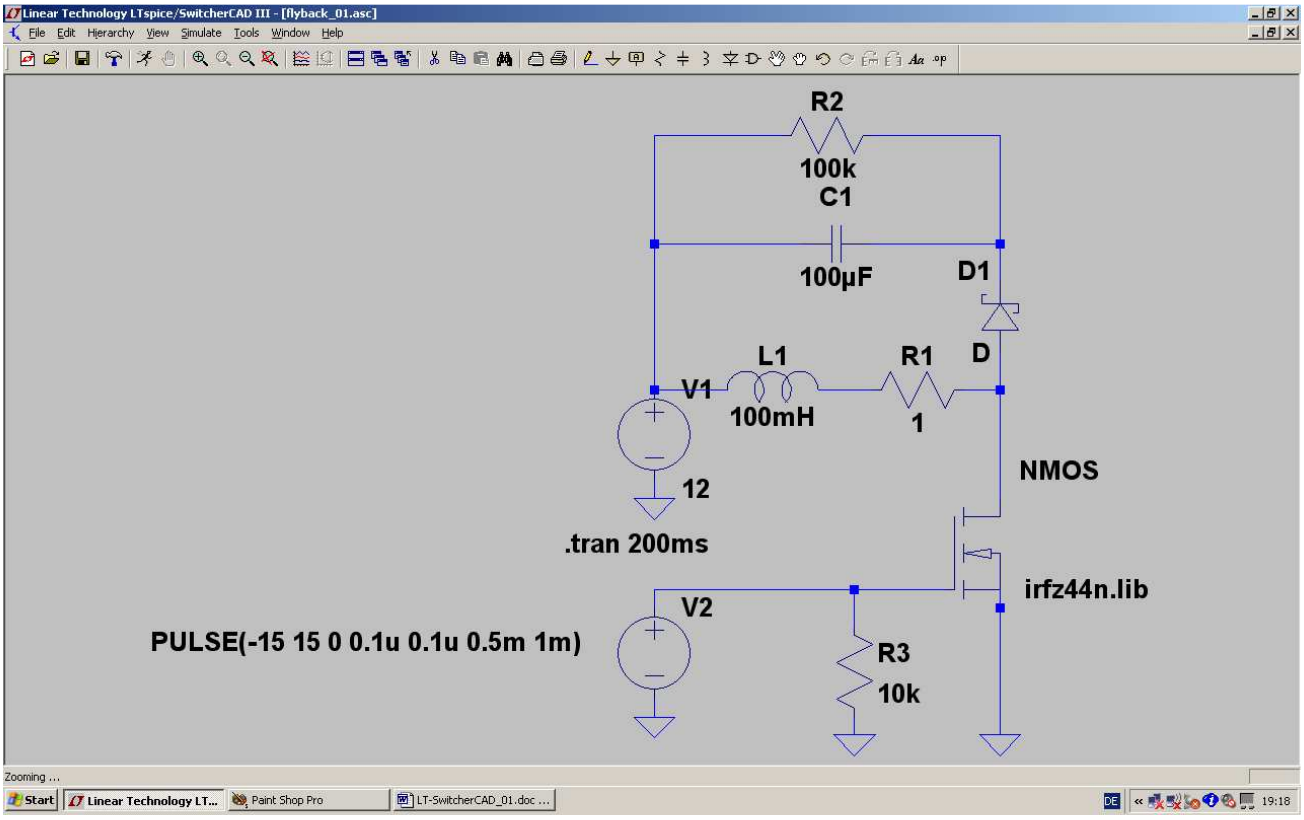
The MOSFET is switched OFF. The Inductor's Voltage raises linearly because the Output Capacitor will be charged up in this time.

dx = 2.88303ms(346.857Hz) dy = 26.4V (slope = 9166.15V/s)

If you want: vary the gate voltage, the inductor's value, the output capacitor's value and simulate....

Note:
In practical circuits the stabilization of the output voltage is done by varying the on to off ratio of the driving squarewave voltage at the gate of the MOSFET.

11.3. The Flyback Converter


Linear Technology LTspice/SwitcherCAD III - [flyback_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
R2
100k
C1
100µF
D1
D
L1
R1
V1
100mH
1
12
NMOS
.tran 200ms
V2
irfz44n.lib
PULSE(-15 15 0 0.1u 0.1u 0.5m 1m)
R3
10k
Zooming ...
Start
Linear Technology LT...
Paint Shop Pro
LT-SwitcherCAD_01.doc ...
DE
19:18

Note:
We have nearly the same schematic -- **with the exception that the resistor-capacitor load is returned to the supply voltage and not to ground!**

When the MOSFET is switched ON, then the inductor current rises again linearly as before. but when the MOSFET is switched OFF, then the current flows through the schottky diode into C1 and R2 to the power supply. This is driven by the stored magnetic energy in L1. This current flow stops when the complete magnetic energy stored in L1 is transferred to C1 and R2.

This circuit has some very interesting properties:

a) You can short circuit this converter at the output terminal without damage. The energy transfer is like a bucket full of water: when filled (= energy stored in the inductor) it cannot transfer more but this portion of water (= energy) to the output. No damage will occur, neither to the MOSFET nor to the other parts of the converter.

b) But be careful: if you run this circuit without the load resistor R2, the transferred energy will add and add and add to the capacitor C1 until a big bang happens! So don't forget this resistor and be sure that its value is always less than **100kΩ.**

c) Note that the output voltage's polarization is now inverted to that of the step-up-converter…..and the output voltage's amplitude is normally much higher than the supply voltage!

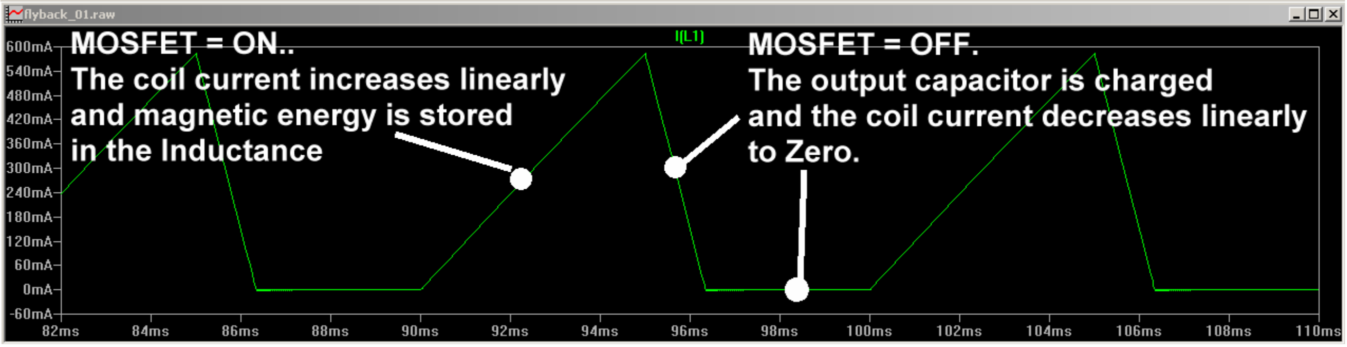
Note:
In practical circuits the stabilization of the output voltage is done by varying the on to off ratio of the driving squarewave voltage at the gate of the MOSFET.

Now draw this schematic and simulate. As gate input voltage again use the symmetric square wave signal with f = 1kHz and +15V / -15V as amplitude.

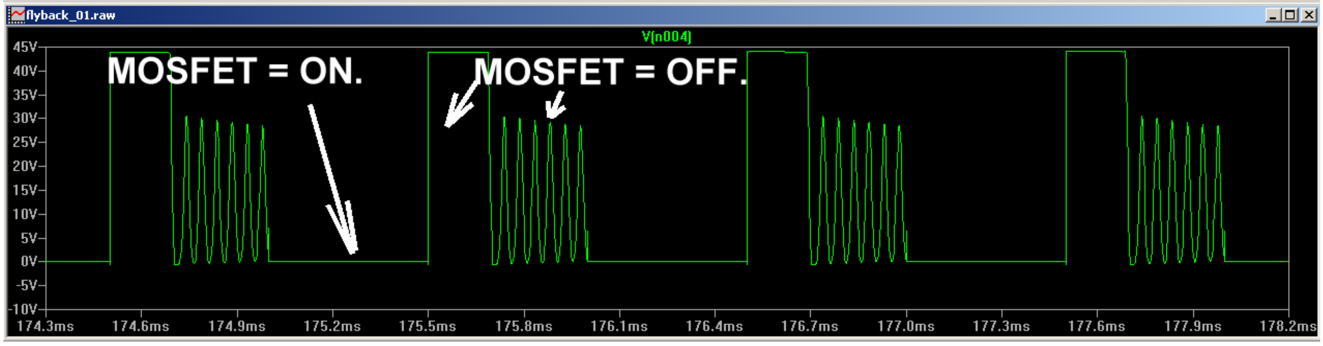
Please have a look at this special effect:

Because the schottky diode's ON-voltage is 0.4V, some energy remains in the inductor L1 when the diode switches OFF. This gives a resonant oscillation caused by the self capacitance of L1 and the drain capacitance of the MOSFET together with the inductance of L1.

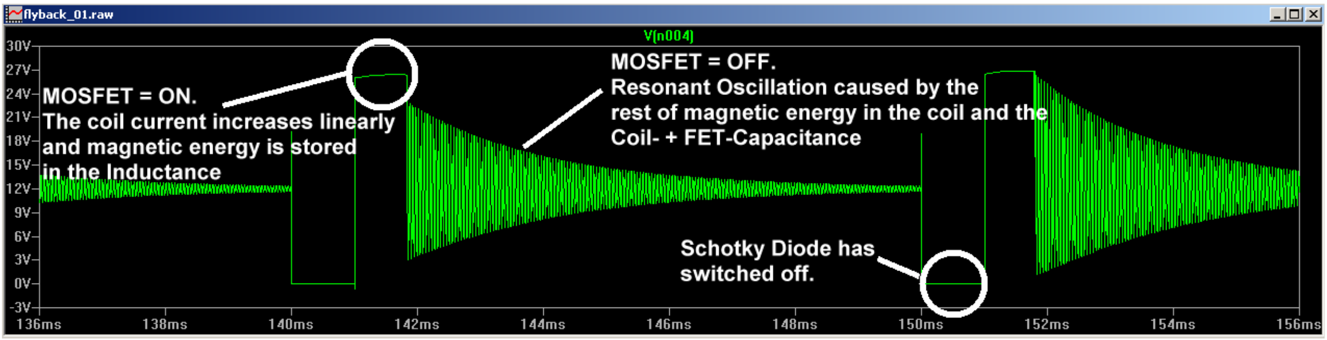
Let's at first look at the inductor current:



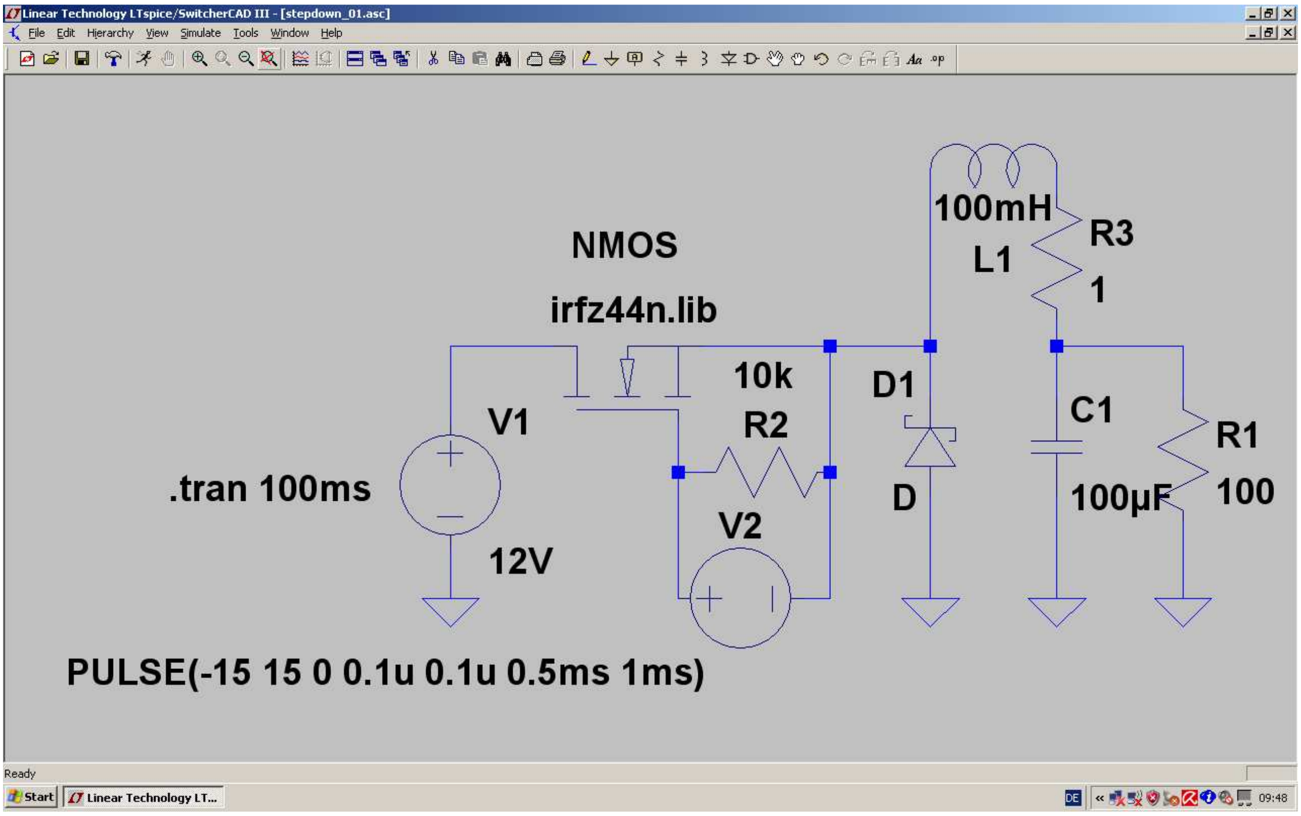
And this is the drain voltage of the MOSFET:



Additional information: this is the simulation with a short ON-Time (1ms) and a long OFF-Time (9ms) of the MOSFET:



11.4. The Step Down Converter


Linear Technology LTspice/SwitcherCAD III - [stepdown_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
NMOS
irfz44n.lib
100mH
L1
R3
1
10k
R2
D1
D
C1
100µF
R1
100
V1
12V
V2
.tran 100ms
PULSE(-15 15 0 0.1u 0.1u 0.5ms 1ms)
Ready
Start
Linear Technology LT...
DE
09:48

In this circuit the MOSFET serves as a switch between the power supply and the inductor and the schottky diode is connected between the node "MOSFET's source / inductor" and ground (initially reverse biased).

Description:

a) When the MOSFET is switched ON then the linear rising current flows from the power supply through the FETand the inductor to the output capacitor and the load resistor to ground. So the capacitor is charged and energy is stored in the magnetic field of the inductor.

b) When the MOSFET is switched OFF the inductor current continues to flow in the same direction, but now the MOSFET is OFF and the schottky diode switches on to maintain current flow. The stored magnetic energy is transferred to the output circuit (= C1 and R1) until the inductor current has decreased to zero. During the time this current flows the voltage across the diode is **NEGATIVE with respect to ground!!!**

Note:

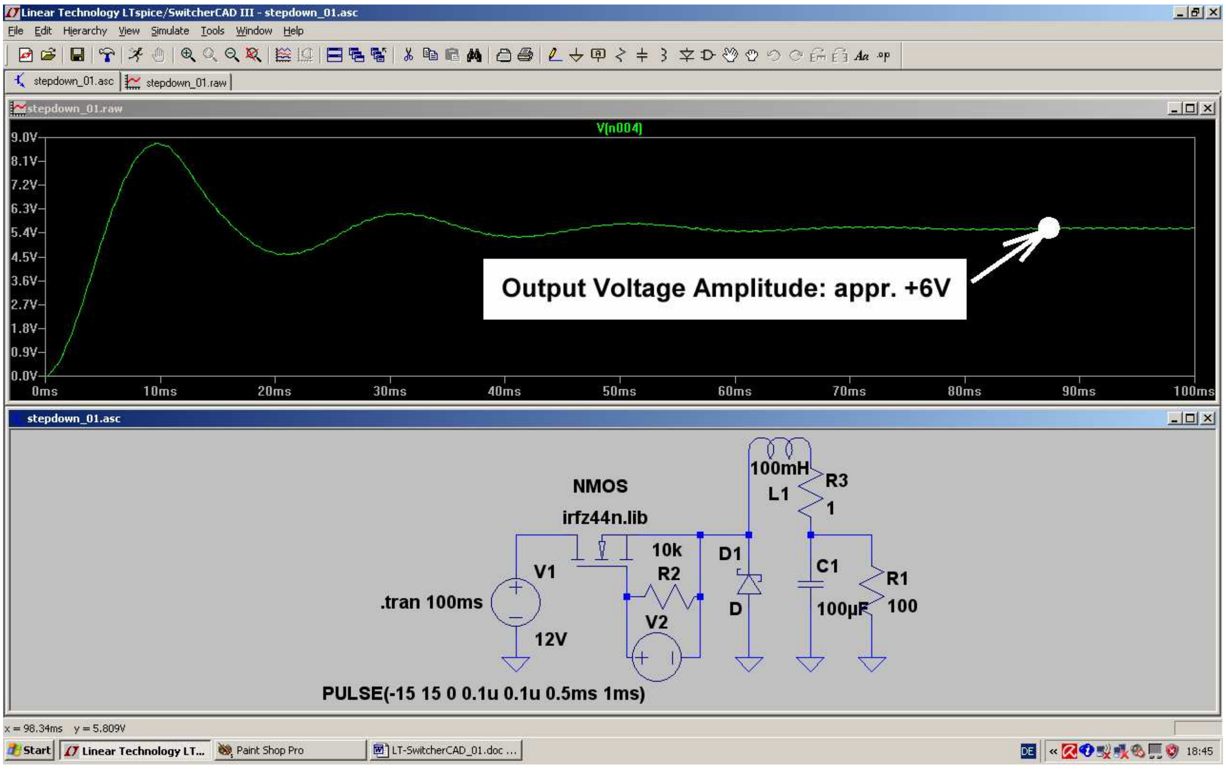
1) In practical circuits the stabilization of the output voltage is always done by varying the on to off ratio of the driving squarewave voltage at the gate of the MOSFET.

2) The amplitude of the output voltage is always less than the power supply voltage!

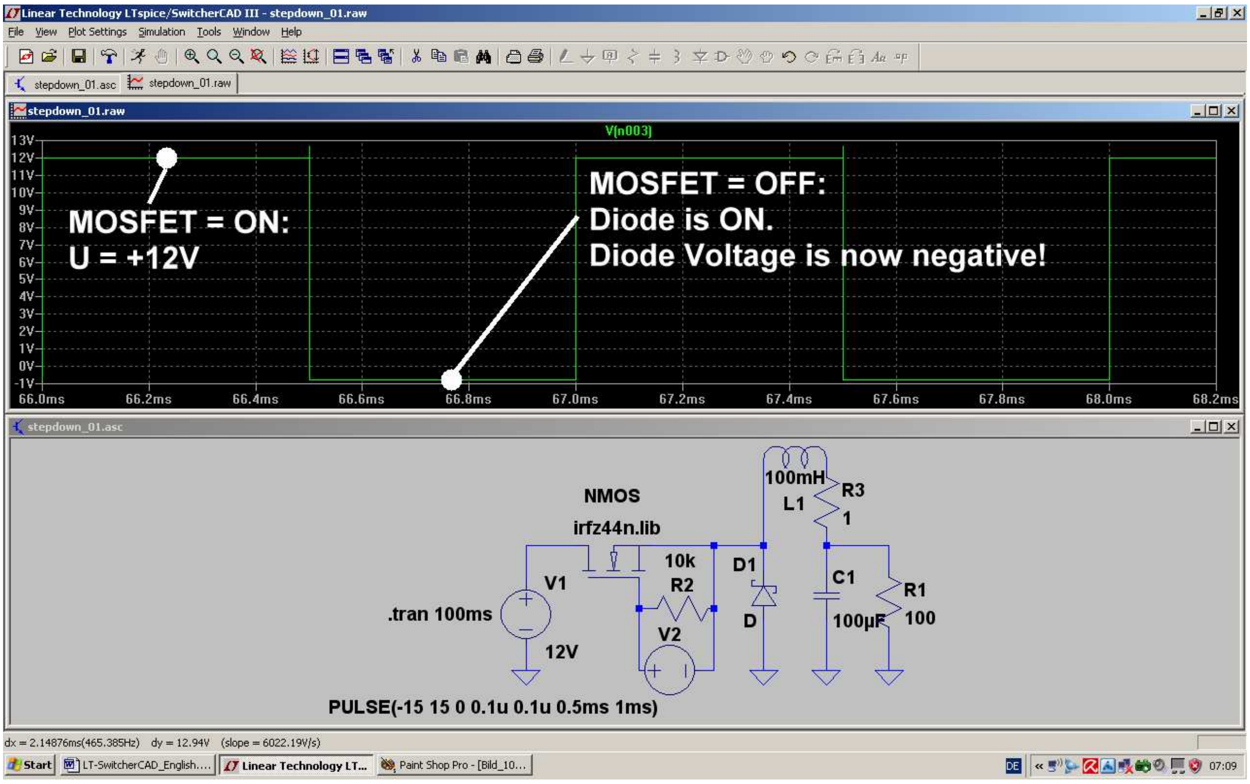
If you want to know the exact output amplitude value, multiply the supply voltage with the ratio „ON-time / period time". So for our example we get with an ON-time of 0,5ms and a period time of 1ms

Vout = 0,5 x 12V = 6V

Let’s first simulate the output voltage:



And this is the diode voltage:


Linear Technology LTspice/SwitcherCAD III - stepdown_01.raw
File View Plot Settings Simulation Tools Window Help
stepdown_01.asc
stepdown_01.raw
V(n003)
13V
12V
11V
10V
9V
8V
7V
6V
5V
4V
3V
2V
1V
0V
-1V
66.0ms
66.2ms
66.4ms
66.6ms
66.8ms
67.0ms
67.2ms
67.4ms
67.6ms
67.8ms
68.0ms
68.2ms
MOSFET = ON:
U = +12V
MOSFET = OFF:
Diode is ON.
Diode Voltage is now negative!
NMOS
irfz44n.lib
100mH
L1
R3
1
10k
R2
D1
D
C1
100µF
R1
100
V1
.tran 100ms
12V
V2
PULSE(-15 15 0 0.1u 0.1u 0.5ms 1ms)
dx = 2.14876ms(465.385Hz) dy = 12.94V (slope = 6022.19V/s)
Start
LT-SwitcherCAD_English....
Linear Technology LT...
Paint Shop Pro - [Bild_10...
07:09

12. Project 8: Thyristor Circuits

12.1. Thyristor Model

A good choice for our experiments is the

2N5171

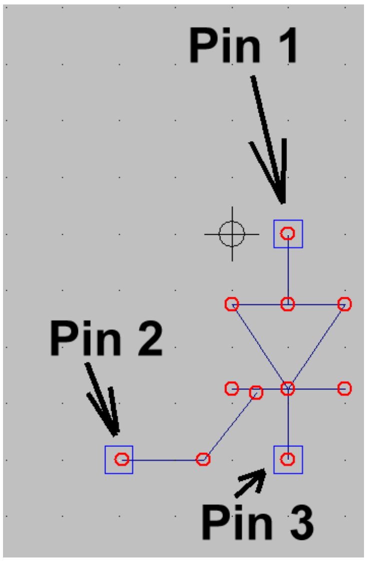
This is a 20A / 600V – thyristor which is used in many circuits. For a SPICE model search in the Internet for the file

thyristr.lib

Then save this library file under **„lib / sub“** **in the LTSpice directory**

But please note:
This library comes as an HTML-file! So open it, select all the text, copy the content to the clipboard and paste it into a new editor file as text. Save this as „thyristr.lib“ in the LTspice library.

And now follow the same procedure as before:

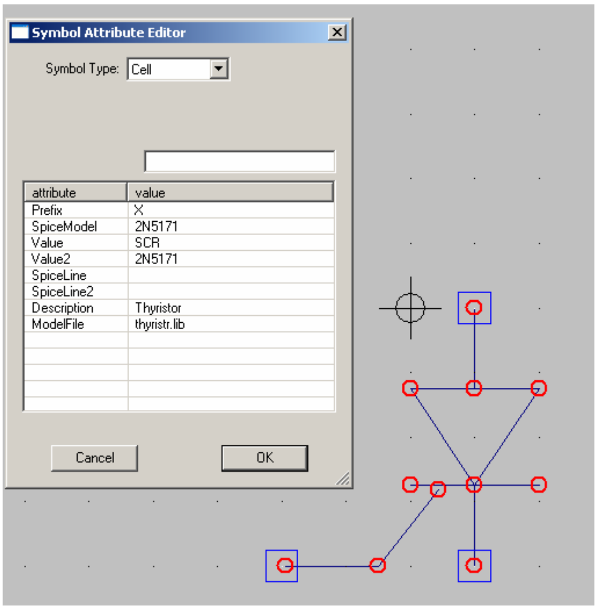


Step 1:
Open „**New Symbol**“ in the file menu. Then open the **symbol for a diode („diode.asy“ in folder „lib / sym).** Then draw the gate pin for a thyristor.

Warning:
The SPICE model uses the following numbering for the pins

Anode	**=**	**pin 1**
Gate	**=**	**pin 2**
Cathode	**=**	**pin 3**

So be absolutely sure that you have the same numbering in your symbol (see left figure). If not, change it.



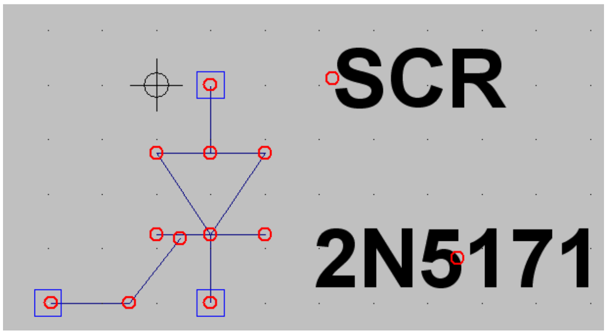
Symbol Type: Cell

attribute	value
Prefix	X
SpiceModel	2N5171
Value	SCR
Value2	2N5171
SpiceLine	
SpiceLine2	
Description	Thyristor
ModelFile	thyristr.lib

Step 2:
Now select from the menu bar „**Edit / Attributes**“ and choose “**Edit Attributes**“

Enter in the list:

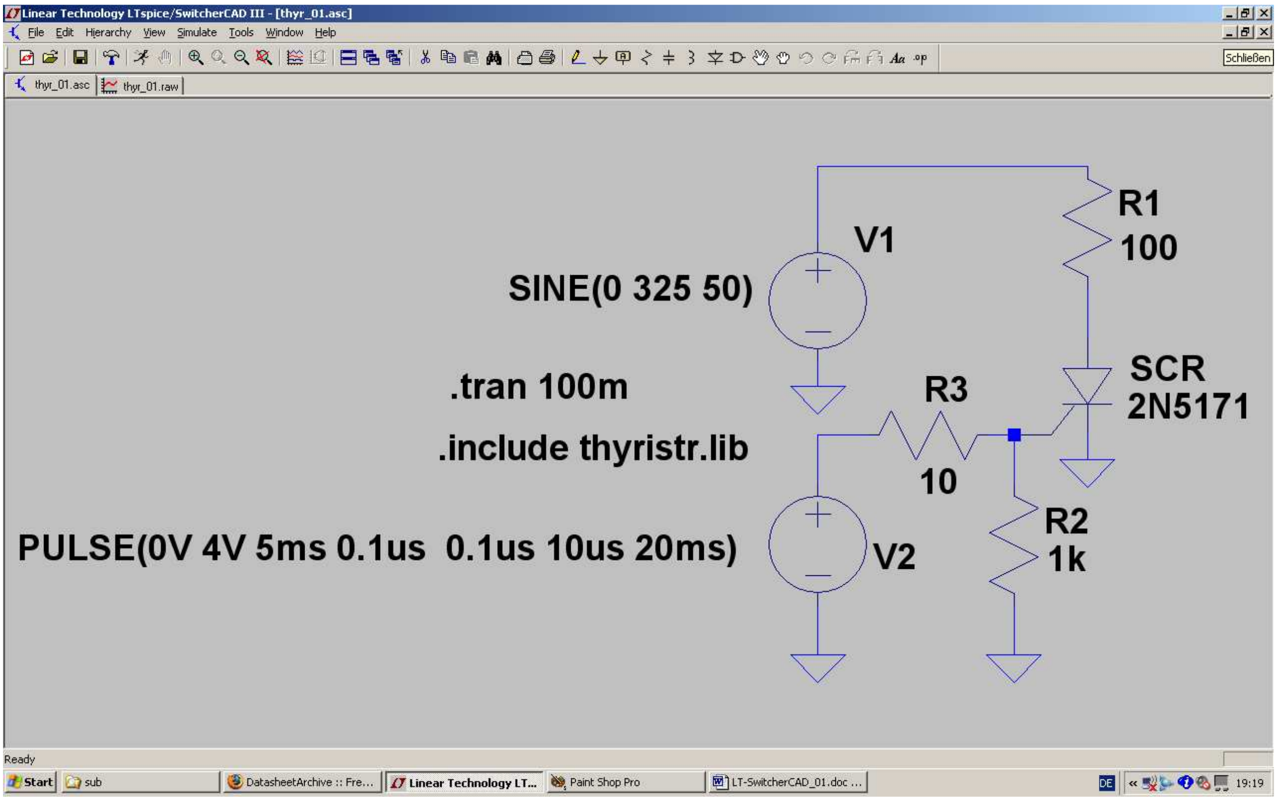
Prefix	**X**
SpiceModel	**2N5171**
Value	**SCR**
Value 2	**2N5171**
Description	**Thyristor**
ModelFile	**thyristr.lib**



Step 3:
Open „**Edit / Attributes**“, but now choose „**Attribute Window**“. Click on „**Value**“ in the list and place “**SCR**“ beside the symbol. Repeat the procedure for „**SpiceModel**“. This should be the result.

Now the finished symbol must be saved in a new folder (named „**Thyristors“)** in **„lib / sym“** using the name **2N5171.asy**“.

12.2. Switching Resistive Loads


Linear Technology LTspice/SwitcherCAD III - [thyr_01.asc]
File Edit Hierarchy View Simulate Tools Window Help
Schließen
thyr_01.asc
thyr_01.raw
V1
R1
100
SCR
2N5171
R3
10
R2
1k
V2
Ready
Start
sub
DatasheetArchive :: Fre...
Linear Technology LT...
Paint Shop Pro
LT-SwitcherCAD_01.doc ...
DE
19:19

SINE(0 325 50)

.tran 100m

.include thyristr.lib

PULSE(0V 4V 5ms 0.1us 0.1us 10us 20ms)

Description:
A sine wave source V1 (peak value = 325 V, frequency = 50Hz) feeds a series connection of a resistor R1 (100 Ω) and the thyristor 2N5171. Between the gate and the source a resistor R2 (1kΩ) is added to the schematic. A pulse source V2 is connected to the gate of the thyristor using a “**current limiting resistor R3 with 10Ω”.**

Explanation of other entries on the schematic:
:

a) **„.tran 100m“** gives a simulation time from 0.....100ms.

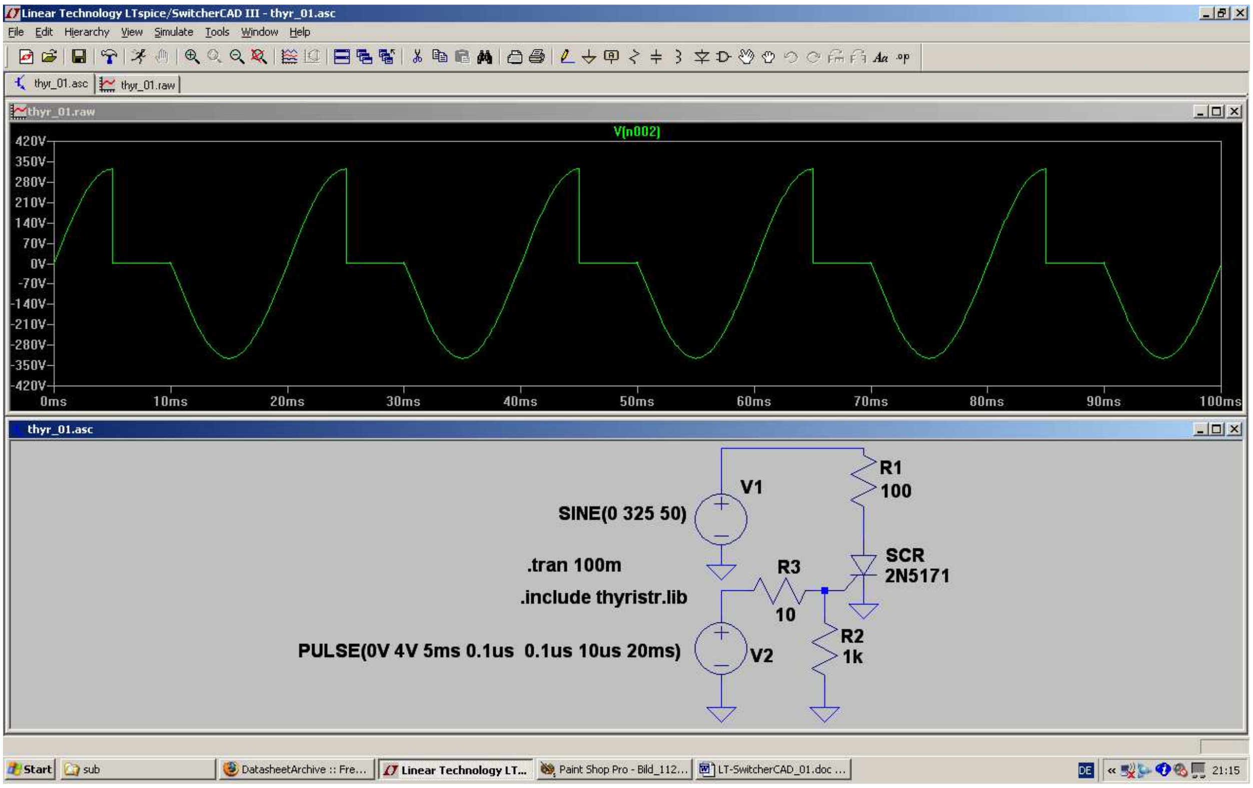
b) „**.include thyristr.lib“** prepares for the usage of the library with the thyristor SPICE-Models.

c) The pulse voltage at the gate is programmed by the line:

PULSE (0V 4V 5ms 0.1us 0.1us 10us 20ms)

This gives a minimum amplitude of 0V and a maximum amplitude of +4V. The start delay time is 5ms, the rise and fall times are 100ns. Pulse length is 10µs with a period of 20ms.

So the anode voltage looks like:



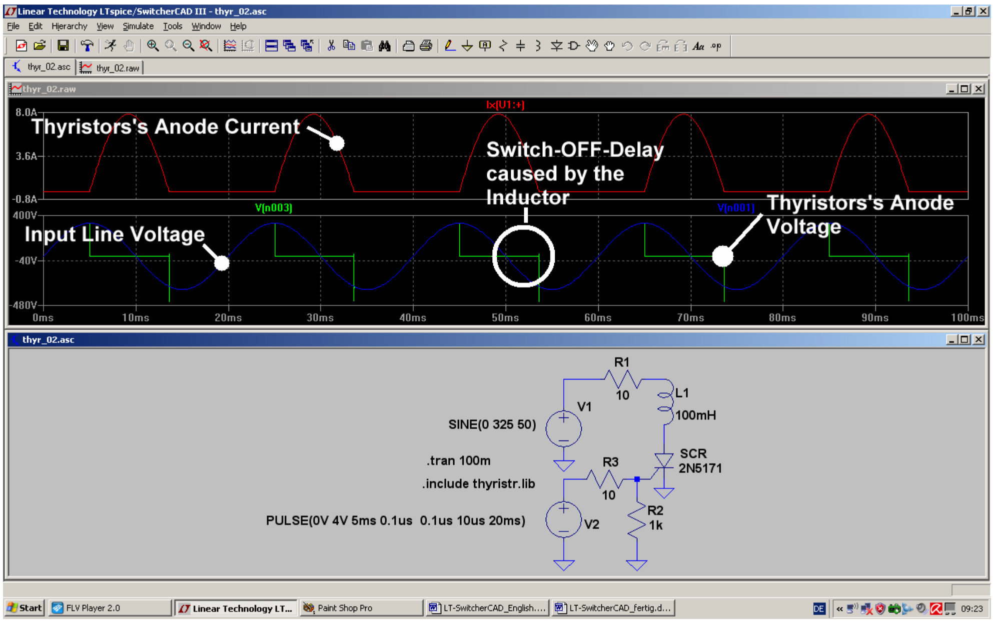
.tran 100m

.include thyristr.lib

PULSE(0V 4V 5ms 0.1us 0.1us 10us 20ms)

12.3. Switching Inductive Loads

This is often a problem due to the stored magnetic energy in the inductor. Every time when the line voltage crosses 0V, the thyristor wants to switch off. But the magnetic energy stored in the coil keeps the current flowing in the same direction -- therefore the thyristor stays ON as long as there exists enough energy to do so …and so you get a „Switch OFF – delay“ of the thyristor’s anode current.



.tran 100m

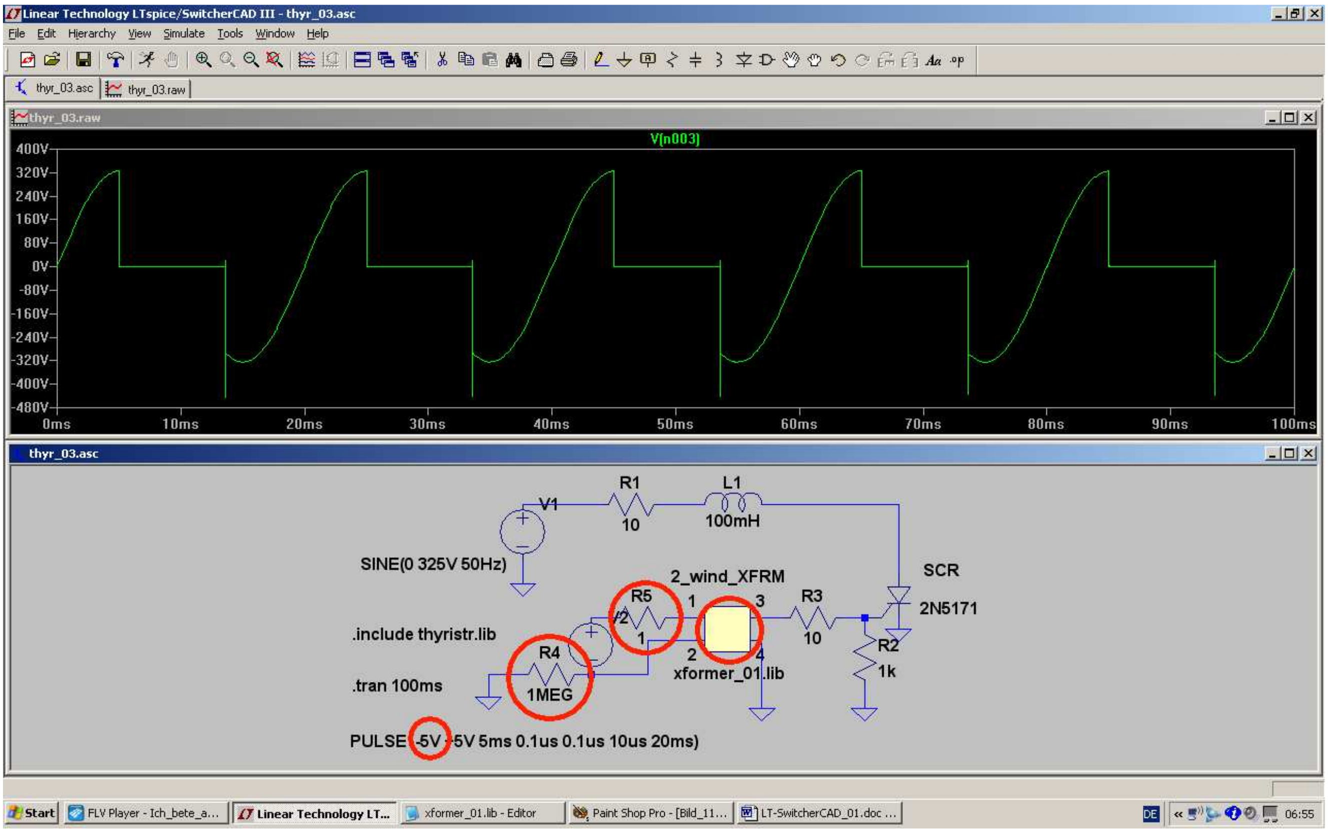
.include thyristr.lib

PULSE(0V 4V 5ms 0.1us 0.1us 10us 20ms)

With an inductor of L =100mH and a series resistor of R = 10Ω we get this simulation result and we see that the thyristor is unable to switch OFF at the line voltage’s zero crossing.
Current will flow as long as there is any remaining magnetic energy stored in the inductor. After the current drops below the holding current of the SCR the small amount of remaining energy gives a short damped oscillation (= coil + drain capacitance = resonant circuit).

12.4. Circuit with Gate Transformer

Very often practical circuits need complete isolation between the load circuit and the gate circuit. So in this case a gate transformer can be used to fire the thyristor:



Some details:

a) The pulse source to trigger the thyristor is now isolated from the gate by the transformer. But SPICE does not accept „floating circuits or nodes“. So resistor R4 (1MEG) has been added to make the necessary ground connection.

b) Also the properties of the pulse voltage must be changed because a transformer cannot transfer DC values from the primary to the secondary winding. So we use “-5V” and “+5V” as minimum and maximum amplitudes.

c) Never connect an inductor without a series resistor to a voltage source……so R5 was added to the primary winding of the transformer.

d) The transformer is our well-known part „**xformer_01**“ of the rectifier experiments.

13. Project 9: Echoes on Transmission Lines

13.1. Transmission Lines -- only two Wires?

When considering a simple electrical circuit consisting of a voltage source and a load, everything seems to be easy: on the upper wire the current flows from the source to the load, on the lower wire the current flows back from the load to the source. But where is there a problem with that?

The circuit must be considered as a transmission line when a source transmits energy to a load that is some distance away. In other words, when the distance is greater than about 1/10th of the wavelength the circuit behaves more like a transmission line. At very high frequencies this can be in the cm or even mm range.

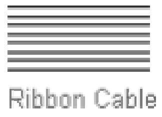
Let us at first have a look at the ways we can construct a transmission line:

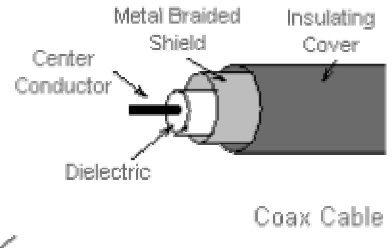
Single Pair

Twisted Pair

The „**Single Pair**" is simple: two wires held in parallel.

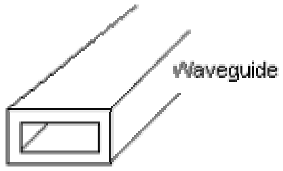
The „**Twisted Pair**" is the well - known standard for a LAN, CAN or telephone lines.



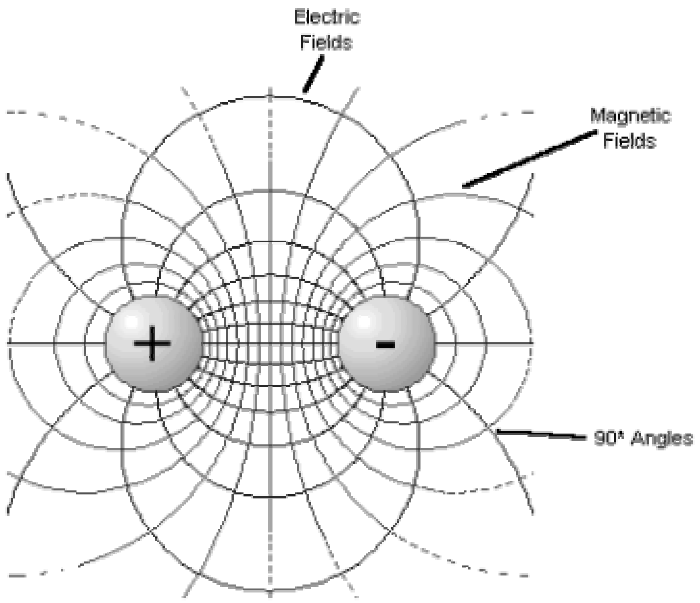


The **Coaxial Cable is widely used in the Communications, Radio, TV and Video.**

You can find „**Ribbon Cable**" used as connection between the PC's mainboard and the hard disk.



An unusual transmission line is the **waveguide**", because an empty space serves as the transport media for the electric and magnetic fields.

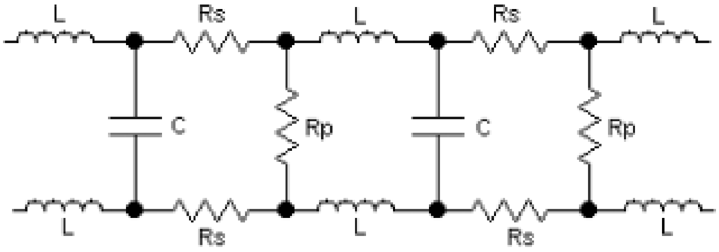


As soon as you apply a voltage to the input of a line (here the single pair is shown) you also get a magnetic field when current flows.
So please take note of the field distribution in the figure on the left.

It is now important to know that the electric field is representative of a capacitor. **This value increases with the line's length!**

As soon as current flows a magnetic field forms around every wire. **This is representative of an inductor and this inductor's value increases with the line length.**

To calculate the properties of the line you cut it in pieces -- but you must take very short pieces to simulate the "distributed capacitors and inductors".



Additionally you have to take into account the losses of the inductors (...caused by the skin effect which increases with frequency..) as series resistors. The losses of the insulation medium are representated by parallel resistors.

So you can extend the simple schematic model for a longer line by repeating the circuit.

As soon as a signal is applied to the input of the line, the following effects can the observed:

a) At the first instance of time the generator sees the cable simply as a resistive load with a value of the „characteristic cable impedance", i.e. 50Ω. The applied voltage at the cable input enters the cable and starts to travel along it (= "incident wave"). This characteristic cable impedance can be calculated as follows:

$$\mathbf{Z} = \sqrt{\frac{\textbf{Inductance}}{\textbf{Capacitance}}}$$

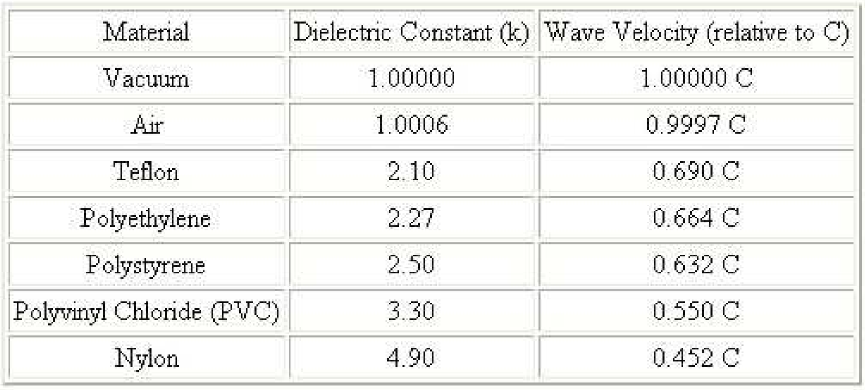
The inductance and capacitance values always refer to a defined cable length (mostly: 1m). For the well known RG58-type you have values of

100pF / 1m, 250nH / 1m

b) When the input signal is travelling along the cable length, the small capacitors are charged or discharged through small series inductors. This takes a finite time and so this gives rise to **„cable velocity"** or **"wave velocity** which is slower than the „ velocity of light in air".
We get:

$$\mathbf{v}_{\textbf{cable}} = \frac{\mathbf{v}_{\textbf{light}}}{\sqrt{\boldsymbol{\varepsilon}_{\mathbf{r}}}}$$

For a coaxial cable ε_r (....in America sometimes: „k") is the dielectric constant for the cables insulation between inner and outer conductor.



Material	Dielectric Constant (k)	Wave Velocity (relative to C)
Vacuum	1.00000	1.00000 C
Air	1.0006	0.9997 C
Teflon	2.10	0.690 C
Polyethylene	2.27	0.664 C
Polystyrene	2.50	0.632 C
Polyvinyl Chloride (PVC)	3.30	0.550 C
Nylon	4.90	0.452 C

RG58 with Z = 50 Ohm uses polyethylene for this purpose. If you look at the table you can see that the wave velocity is reduced to 66.6% of the velocity of light, "C".

c) But what happens to the travelling energy on the cable? (see following chapter).

13.2. Echoes (= Reflections)

At high frequencies it is difficult to measure currents with high precision. Also, the determination of a source resistance (by “Open Loop” and “Short Circuit” Measurements) is nearly impossible. So we have to change our thinking at frequencies higher than 1MHz.

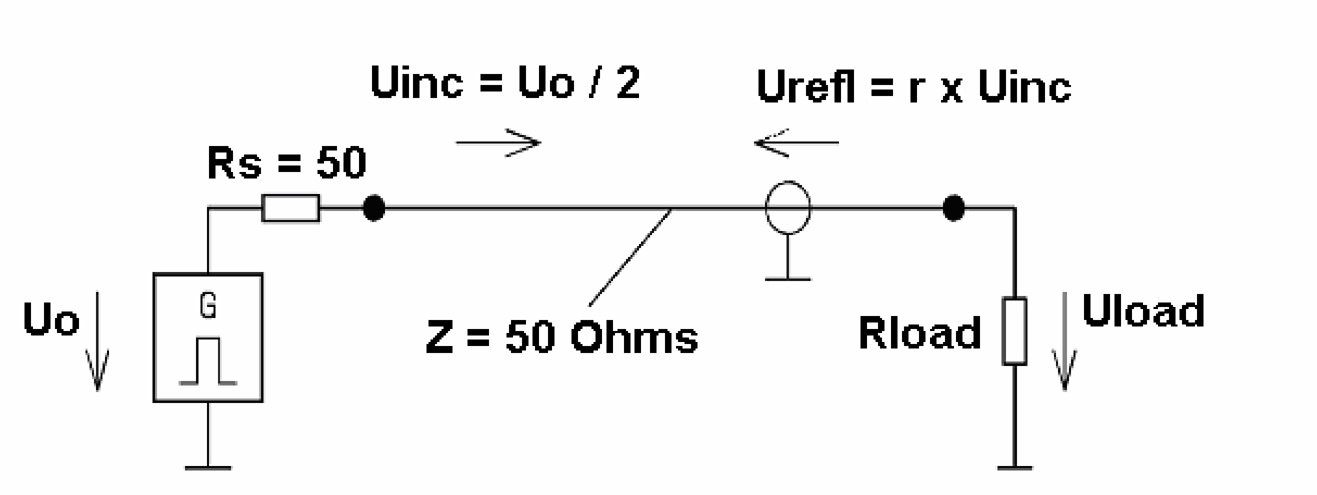
Everywhere in a system we use the same „system impedance“ (75Ω for radio and TV and video, but 50Ω in all other applications). This value is valid for the input- and output Impedance of all used blocks, for the cable impedance and for all terminations in the system.

The reason is very simple: now everywhere in the system you get perfect power matching (Ri = Ra).

With “directional couplers” the differences between ideal and practical situation can be measured and expressed as

Reflection coefficients.

Let us have a look at a pratical example:
A pulse voltage source with a source resistance Rs = 50Ω is connected to a transmission line (coaxial cable RG58 with Z = 50Ω) and generates a short pulse which is sent to the termination resistor Rload at the end of the cable.



Now the following happens:

a) When the cable is very long and the pulse is very short, then the source does not sense the load at the end of the cable. (....You know: the speed of light is 30cm per Nanosecond in the air and 20cm per Nanosecond in a RG58 cable).

b) So the cable’s input resistance is **50Ω** and forms a voltage divider with the source resistance of **50Ω**. This gives power matching and a pulse with amplitude of U0 / 2 enters the cable. The associated power (P = Uo x Uo / 4 x 50Ω) starts to travel at cable speed to the load resistance. This power is called **“incident wave”**

c) When the incident wave arrives at the load, then it can only be absorbed totally if the load resistor has also a value of **50Ω.** Any difference from 50**Ω** causes a mismatch of „perfect power matching“ and so the „power surplus“ is reflected and runs back to source (..at cable speed) as “**reflected wave**”

For the exact description of these effects the **reflection coefficient „r“** was introduced:

$$\mathbf{r} = \frac{\mathbf{U_{reflected}}}{\mathbf{U_{incident}}} = \frac{\mathbf{Z_{Load} - Z}}{\mathbf{Z_{Load} + Z}}$$

Now the reflected wave can be calculated as

$$\mathbf{U_{reflected}} = \mathbf{r \bullet U_{incident}}$$

The voltage U_{Load} at the load resistor is then:

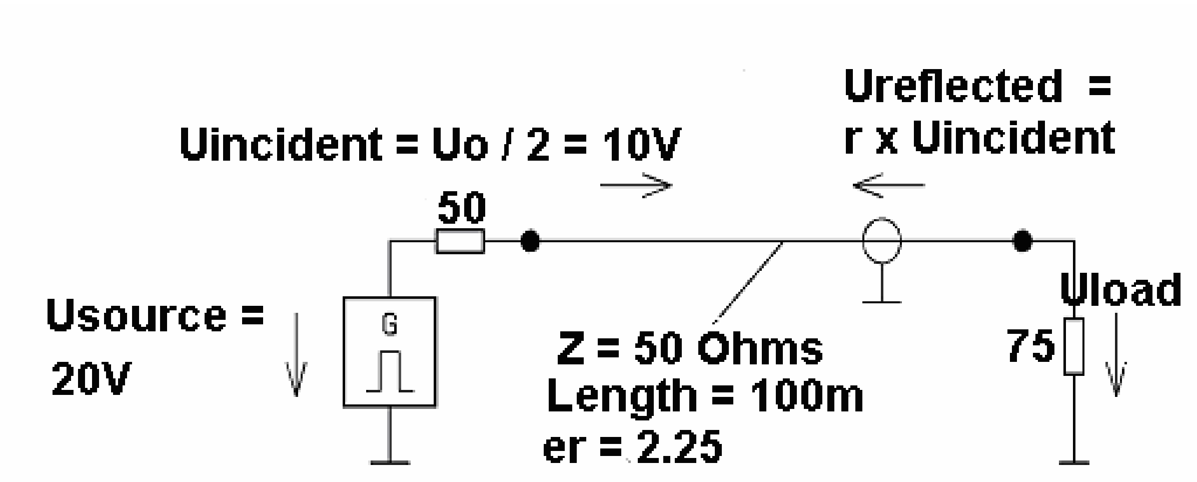
$$\mathbf{U_{Load}} = \mathbf{U_{incident}} + \mathbf{U_{reflected}}$$

Note:
At every point in the cable Ohm’s law must be valid, because you have travelling energies. So you can calculate the associated current at any point:

$$\mathbf{I_{incident}} = \frac{\mathbf{U_{incident}}}{\mathbf{Z}} \quad \text{and} \quad \mathbf{I_{reflected}} = \frac{\mathbf{U_{reflected}}}{\mathbf{Z}}$$

Example:
A pulse generator (Rs = 50Ω) generates in „no load condition“ a pulse amplitude of +20V with a pulse width of 10ns. The pulse repetition frequency is 1kHz. The generator is connected to a RG58-coaxial cable (Z = 50Ω, length = 100m). The cable is terminated by a 75Ω-resistor. The dielectric constant of the cable is er = 2.25.

Calculate and draw the signals
a) At cable’s input b) at cable’s centre c) at cable’s end



Solution:

a) Calculation of the reflection coefficient: $$\mathbf{r} = \frac{\mathbf{Z_{Load} - Z}}{\mathbf{Z_{Load} + Z}} = \frac{\mathbf{75\Omega - 50\Omega}}{\mathbf{75\Omega + 50\Omega}} = +\mathbf{0.2}$$

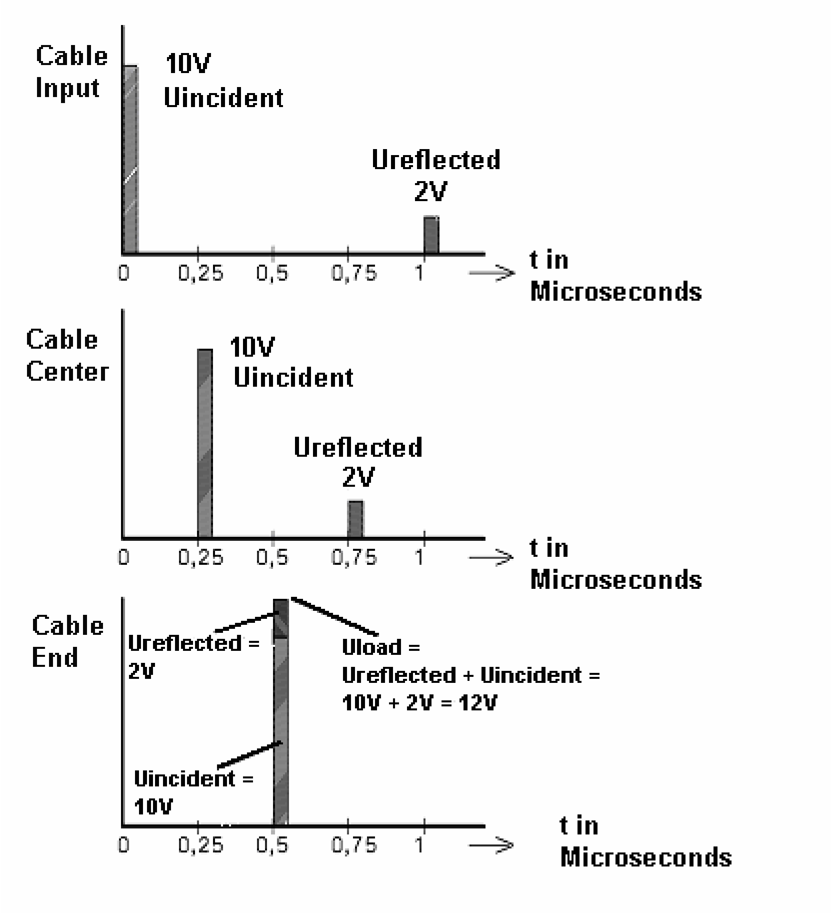
b) Cable velocity: $$\mathbf{v_{Cable}} = \frac{\mathbf{c}}{\sqrt{\boldsymbol{\varepsilon}_{\mathbf{r}}}} = \frac{\mathbf{3 \bullet 10^8 m}}{\sqrt{\mathbf{2.25}} \bullet \mathbf{s}} = \mathbf{2 \bullet 10^8 \frac{m}{s}}$$

c) Signal runtime for 100m of cable length: $$t_{run} = \frac{l_{Cable}}{v_{Cable}} = \frac{100m \bullet s}{2 \bullet 10^8 m} = 0.5 \bullet 10^{-6} s$$

d) Amplitude of incident wave: $$U_{incident} = \frac{U_{Source}}{2} = \frac{20V}{2} = 10V$$

e) Amplitude of reflected wave: $U_{reflected} = r \bullet U_{incident} = 0.2 \bullet 10V = 2V$

f) Amplitude of load voltage: $U_{Load} = U_{incident} + U_{reflected} = 10V + 2V = 12V$



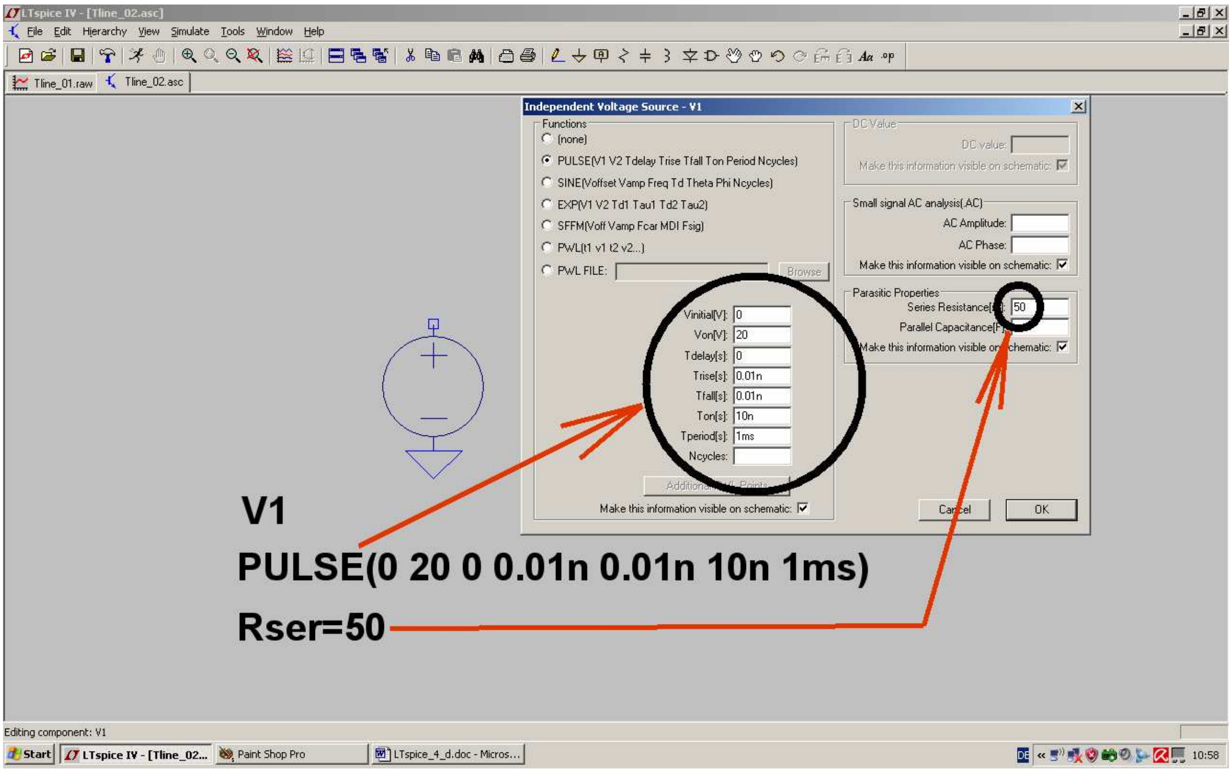
At the cable input we see at first the incident wave (produced by the pulse generator) with an Amplitude of +10V.
After twice the signal runtime for 100m (= 1 Microsecond) the reflected wave comes back from the load and reaches the cable input.

After 0.25 microseconds the incident wave reaches the centre of the cable (= 50m). The echo from the output comes 0.5 microseconds later and passes the centre when travelling back to the cable input.

Exactly after 0.5 microseconds the incident wave reaches the load resistor. Because there is no perfect power match, a part of the arriving energy is reflected and so a reflected wave is generated. At the load resistor we measure the sum of incident and reflected wave = 10V + 2V =12V.

13.3. Simulation of this Example with LTspice

We start with the voltage source from the part library. After placing the symbol on the screen we enter the properties to get a **pulse voltage with a source voltage of 20V, a rise and fall time of 0.01ns, a pulse length of 10ns and a period time of 1 ms. The source resistance must be set to 50Ω:**



Independent Voltage Source - V1

Functions

- ○ (none)
- ● PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)
- ○ SINE(Voffset Vamp Freq Td Theta Phi Ncycles)
- ○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)
- ○ SFFM(Voff Vamp Fcar MDI Fsig)
- ○ PWL(t1 v1 t2 v2...)
- ○ PWL FILE:

Vinitial[V]: 0

Von[V]: 20

Tdelay[s]: 0

Trise[s]: 0.01n

Tfall[s]: 0.01n

Ton[s]: 10n

Tperiod[s]: 1ms

Ncycles:

Make this information visible on schematic: ☑

Parasitic Properties

Series Resistance[Ω]: 50

Make this information visible on schematic: ☑

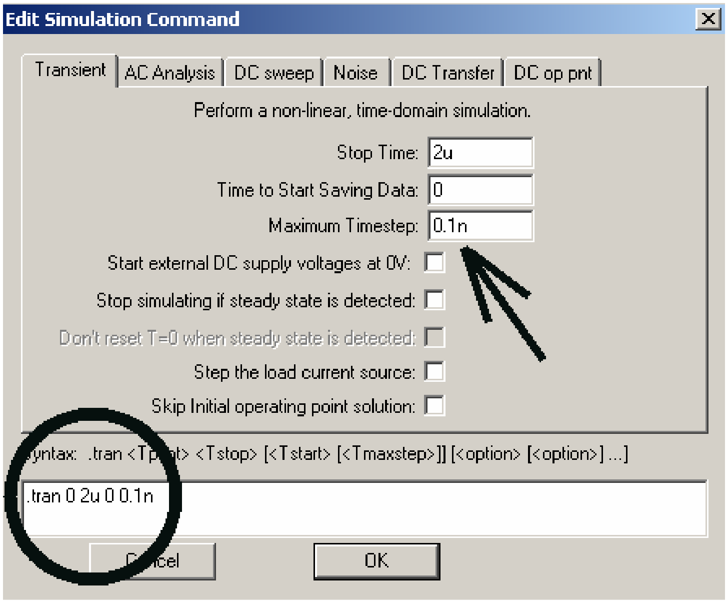
V1

PULSE(0 20 0 0.01n 0.01n 10n 1ms)

Rser=50

Editing component: V1

LTspice IV - [Tline_02.asc]
File Edit Hierarchy View Simulate Tools Window Help
Tline_01.raw
Tline_02.asc
DC Value
DC value:
Make this information visible on schematic:
Small signal AC analysis(.AC)
AC Amplitude:
AC Phase:
Make this information visible on schematic:
Parallel Capacitance[F]:
Browse
Additional PWL Points
Cancel
OK
Start
LTspice IV - [Tline_02...
Paint Shop Pro
LTspice_4_d.doc - Micros...
10:58



Edit Simulation Command

Transient

Perform a non-linear, time-domain simulation.

Stop Time: 2u

Time to Start Saving Data: 0

Maximum Timestep: 0.1n

Start external DC supply voltages at 0V: □

Stop simulating if steady state is detected: □

Don't reset T=0 when steady state is detected: □

Step the load current source: □

Skip Initial operating point solution: □

Syntax: .tran <Tprint> <Tstop> [<Tstart> [<Tmaxstep>]] [<option> [<option>] ...]

```
.tran 0 2u 0 0.1n
```

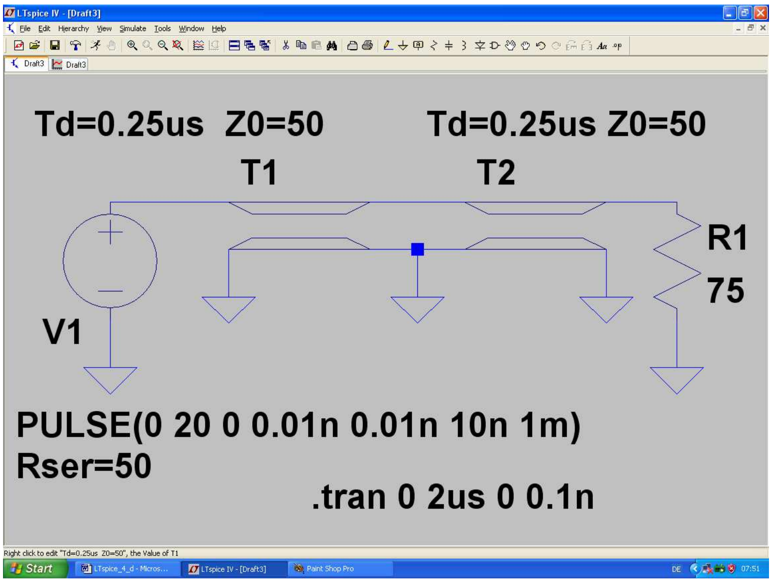
Note:
For such a short rise and fall time of 10 picoseconds it is necessary to use a maximum step size of 100 picoseconds in the simulation settings!

Now we need the **transmission line „tline"** from the part library. But we must enter the "**signal delay time"** in the properties instead of the cable length!

This delay time can be calculated with the mechanical cable length of 100m and the "cable signal velocity" of 200 000 km / sec as follows:

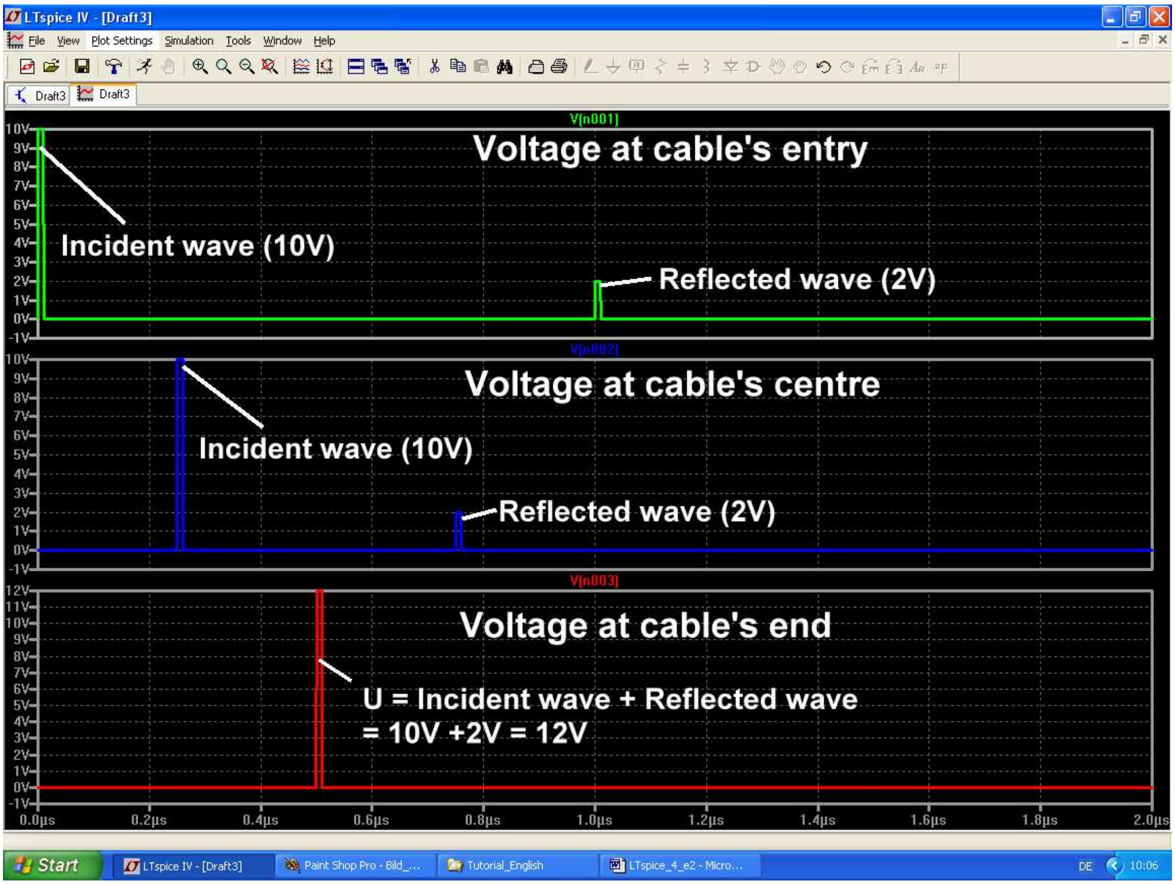
$$t_{delay} = \frac{l}{v_{cable}} = \frac{100m \bullet s}{2 \bullet 10^8 m} = 0{,}5\mu s$$

This delay time of 0.5µs is realized by using two equal pieces of RG58-cable which are connected in series. Each piece produces a delay of 0.25µs and so we get again a total delay of 0.5µs as desired. But now it is possible to view the signals in the MIDDLE of the total cable length.



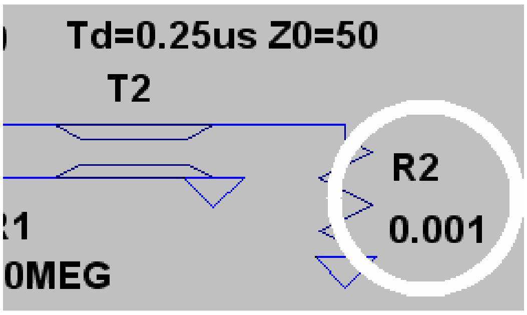
We terminate the end of the cable by a 75Ω resistor and simulate a run time of 2ms (…please check all of the schematic before starting the simulation…):

After the simulation the three voltages (input, middle and output) are presented in three different diagrams. Compare the result with chapter 13.3.:



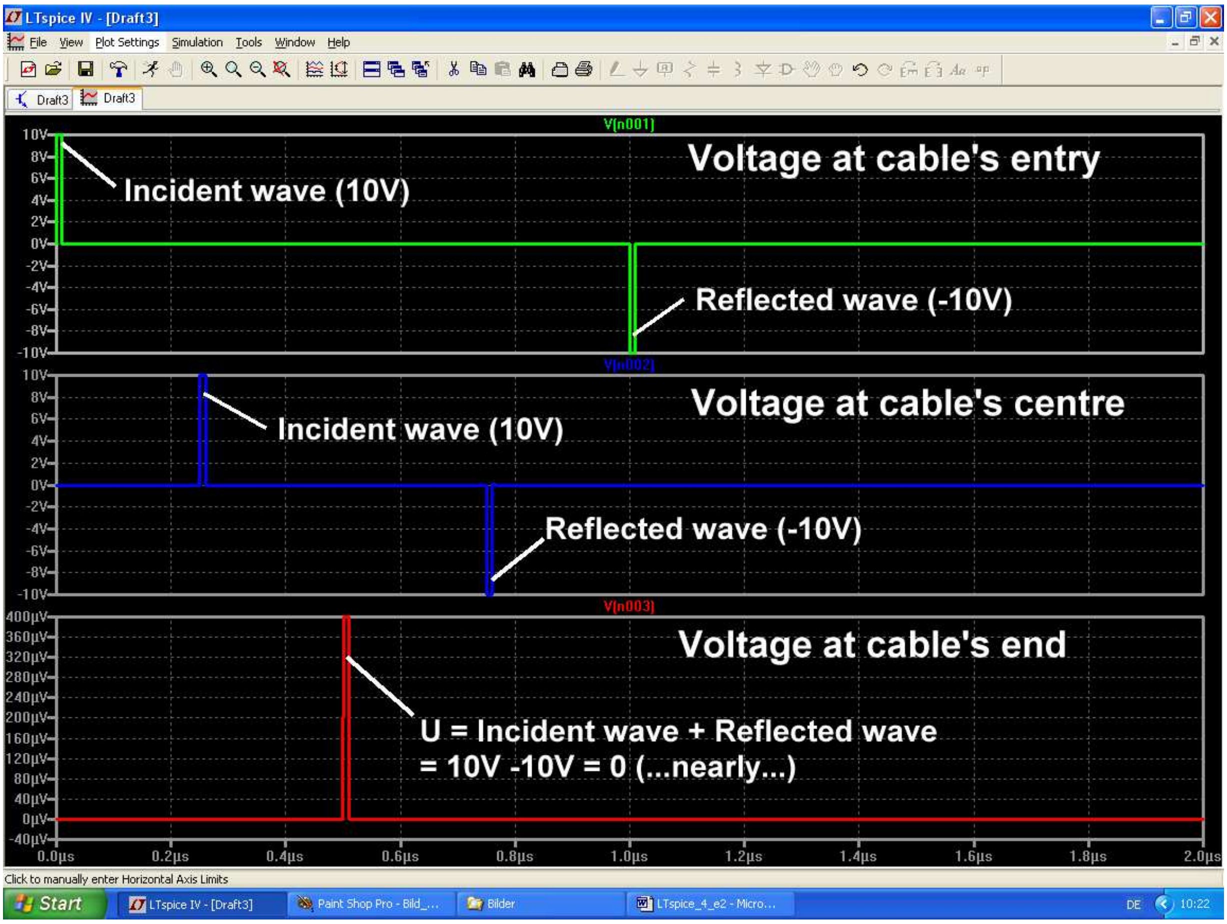
13.4. Open or Short Circuit at Cable’s End

This is a very simple task because it is only necessary to change the value of the termination resistance.



a) **Short Circuit at cable’s end:**
Choose a realistic value of 0.001Ω (= 1 milli-Ohm) as termination.

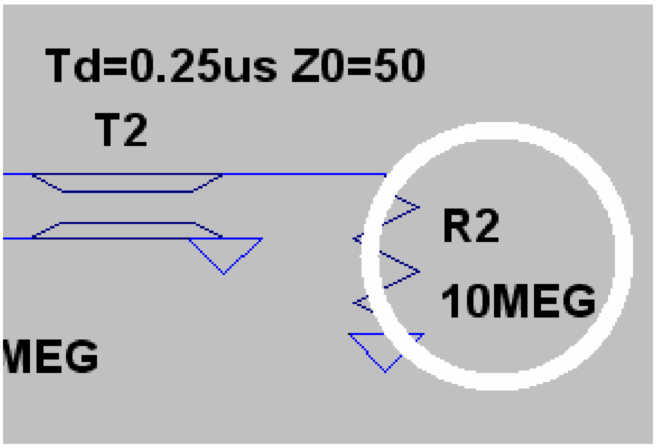
Then you can watch the following signals.



Explanation:
At the short circuit (at cable’s end) the voltage must be zero. This means that no power is delivered into this short circuit and the incident wave (= power travelling from the source to the load along the cable) is sent back to the generator. But to produce zero volts at the short circuit the “reflected wave” (= voltage and power which run back) must change its polarity and so we see a negative echo on the cable.

Because we have replaced the ideal short circuit by a 0.001Ω - termination we get a short pulse with an amplitude of 400 microvolts at the output of the cable.

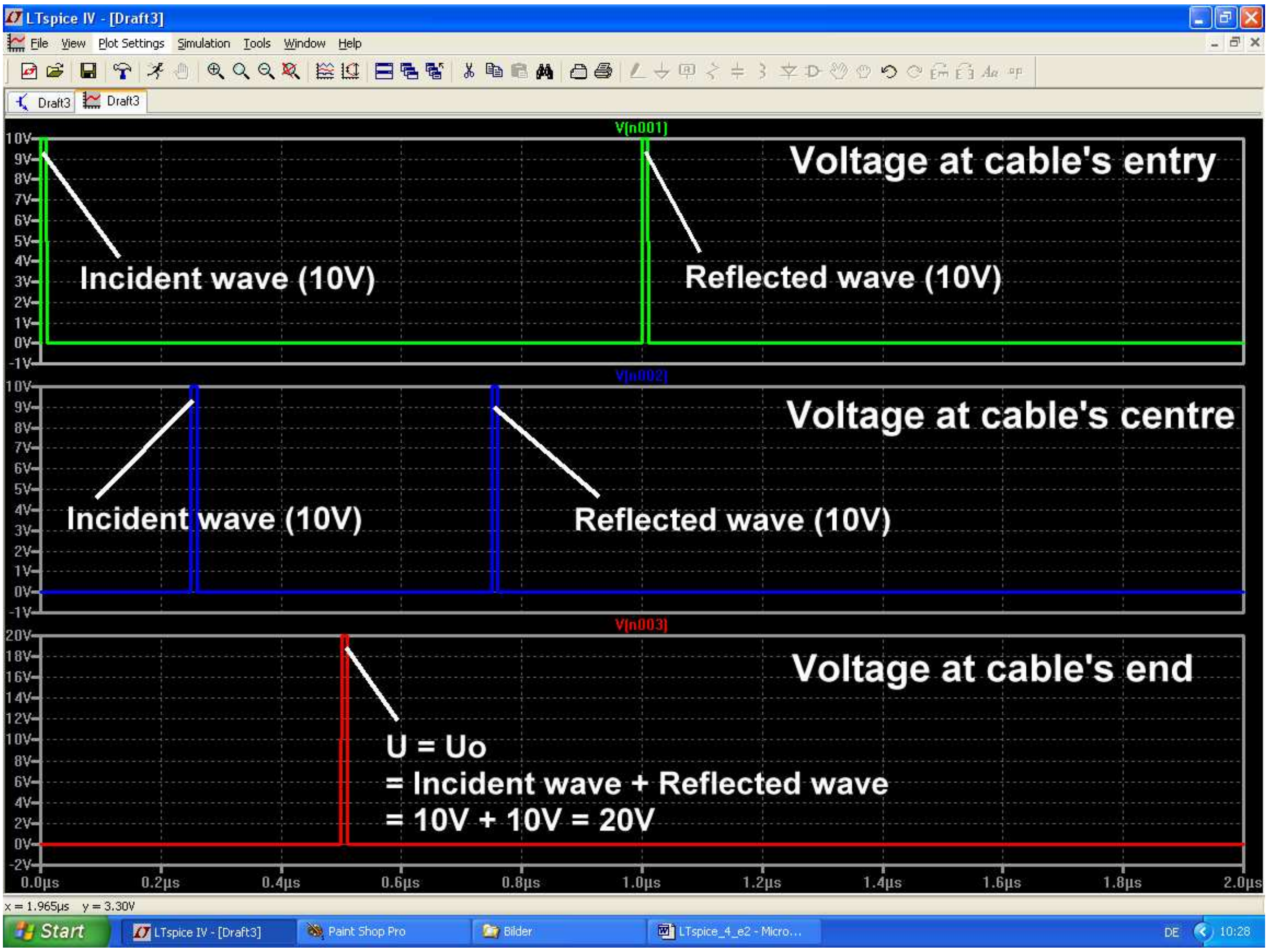
b) Open circuit at cable’s end



Change the termination to 10MΩ -- this will do for an ideal open circuit.

Warning:
Never delete the termination R2 from the schematic to get an ideal open circuit. For the program this would be a “floating node” and at once you get an error message and a simulation abort!

And this is what you get as a simulation result:



LTspice IV - [Draft3]
File View Plot Settings Simulation Tools Window Help
Draft3 Draft3
V(n001)
Voltage at cable's entry
Incident wave (10V)
Reflected wave (10V)
10V 9V 8V 7V 6V 5V 4V 3V 2V 1V 0V -1V
V(n002)
Voltage at cable's centre
Incident wave (10V)
Reflected wave (10V)
V(n003)
Voltage at cable's end
U = Uo
= Incident wave + Reflected wave
= 10V + 10V = 20V
20V 18V 16V 14V 12V 10V 8V 6V 4V 2V 0V -2V
0.0µs 0.2µs 0.4µs 0.6µs 0.8µs 1.0µs 1.2µs 1.4µs 1.6µs 1.8µs 2.0µs
x = 1.965µs y = 3.30V
Start LTspice IV - [Draft3] Paint Shop Pro Bilder LTspice_4_e2 - Micro...
DE 10:28

Note:
Because at the cable’s end the current is zero no power is delivered to the termination. So the incident wave is reflected and runs back to the generator. But now -- as always when a power supply doesn’t have a load -- the voltage at cable’s end rises to the value of the unloaded source voltage (= 20V in our example).

13.5. Lossy Cables (i. e. RG58 / 50Ω)

13.5.1. How can I simulate an RG58 Coaxial Cable?

We need a part from the LTspice – part - library which is named

ltline

(= **lossy transmission line**).

But there is a little problem:

We have to write a special model file with the RG58 properties. These properties can be found in the internet and must be applied to create the desired cable model and symbol.

In the LTspice manual you will find the necessary information which is as follows:

```
.model   RG58   LTRA(len=100   R=1.5   L=250n   C=100p)
```

Details:

„model“ is the SPICE syntax for a part model

„RG58“ is the name of the new part

„LTRA“ means „lossy transmission line“

„len=100“ means the „unit length” and „len = 100“ says that we have a cable length of 100m if the unit is “1m”.

„R=1.5“ means that for 1 unit length (= 1m) a series resistance of 1.5Ω must be taken into account.

„L=250n“ gives an inductance of 250nH per unit length (here: for 1m)

„C=100pF“ gives a capacitance of 100pF per unit length (here: for 1m)

Write this line with an editor and save it as „**RG58.mod**“ using the path „**LTspiceIV / lib / sub**“.

(The properties „100pF per 1m“ and „250nH per 1m“ come from the internet for RG58.
The value for the losses with „1,5Ω per 1m“ is estimated and will be checked in the next chapter……

13.5.2. Simulation of Cable Loss at 100MHz

At first an overview from the internet:

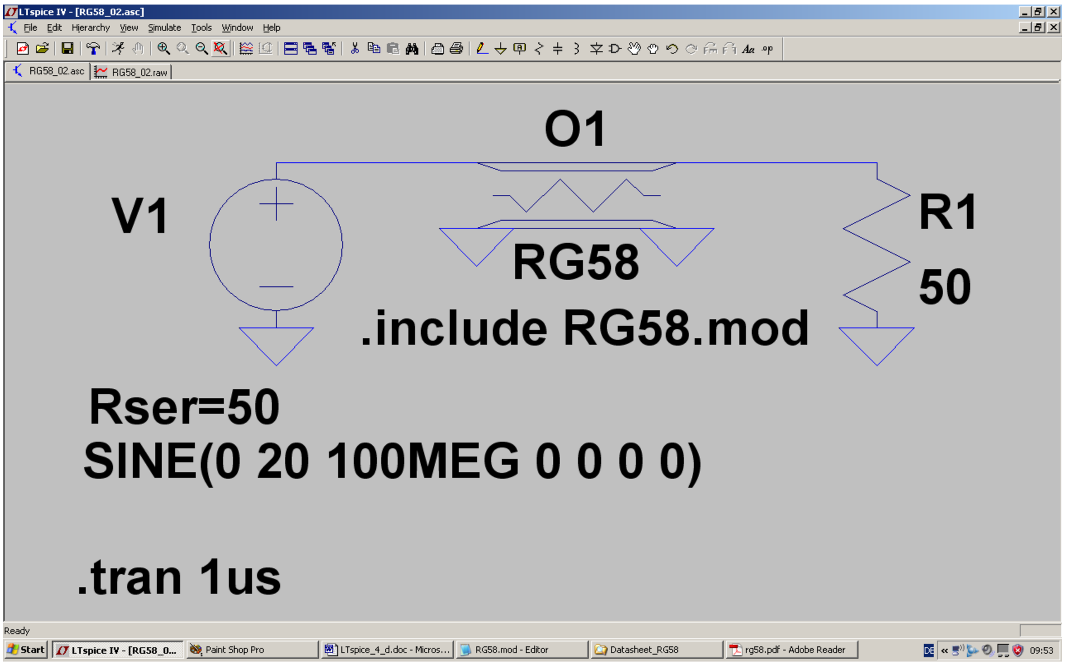
Frequenz *frequency* MHz	Dämpfung typ. Meßwerte *attenuation* *typ. measured values* dB/100m	max. Dämpfung *max attenuation* dB/100m	min. Rückfluß- dämpfung *return loss* dB	$P_{40°C,max}$ W	$U_{40°C,max}$ V
50	10,1	11	20	210	105
100	14,6	16		145	90
200	21,6	24		95	75
300	27,5	30		75	65
450	36,0	40		55	55
500	38,1	42		55	55
800	52,8	58		40	45
900	57,5	64		35	45
1000	62,3	70		30	40
1800	102,4	113		20	30
2000	108,5	120		20	30

As the table shows, losses increase with frequency. So you must prepare a new simulation for each frequency which you want to use.
Let us test this at 100 MHz.

Step 1:
At 100MHz the table gives a value of 14.6dB per 100m. So the output voltage at cable's end for this length is attenuated by the factor

$$10^{\frac{-14,6dB}{20dB}} = 0,186$$

Step 2:
We use a RG58 cable length of 100m (…entered in the RG58.mod – file!) and a termination of 50Ω.
The voltage source works with a sine wave (peak value = 20V) and a source resistance of 50Ω. So we have this schematic:



```
.include RG58.mod
Rser=50
SINE(0 20 100MEG 0 0 0 0)
.tran 1us
```

Note:

a) Right click on „**LTRA**“ under the symbol of the transmission line and replace it by „**RG58**“-

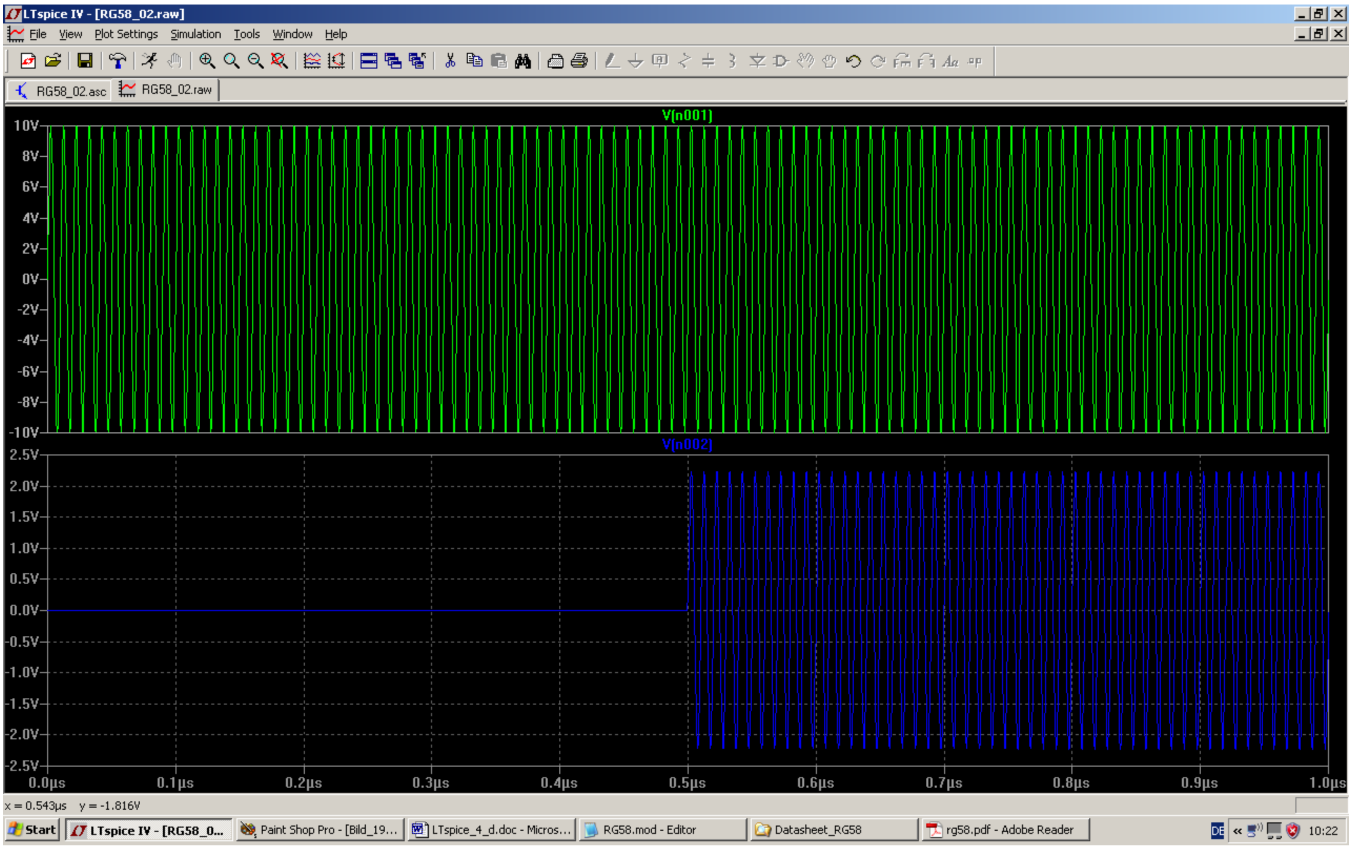
b) Use „**Edit“ and „Spice Directive**“ to get

.include RG58.mod

on the screen..

c) Use a **simulation time from 0….1µs.**

Step 3:
After the simulation use two different diagrams to present the input and the output voltage:

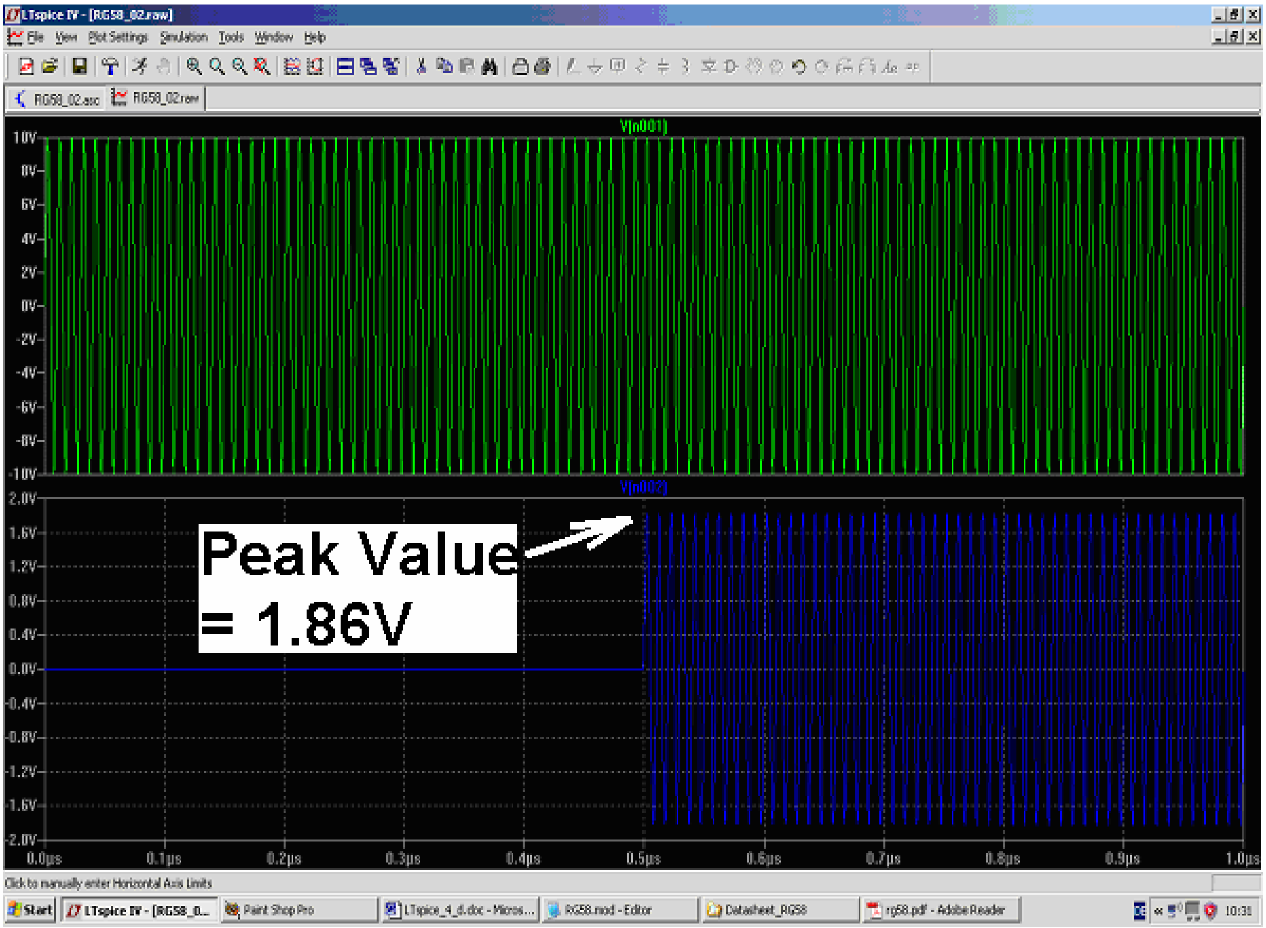


It is nice to determine the „signal propagation time on the cable“ as a starting time difference between the two voltages for a cable length of 100m.

An attenuation of 14.6dB per 100m would decrease the output voltage to 1.86V, but we find 2.25V.
So the series resistance per unit length must be **increased from 1.5 to 1.7Ω** to see the correct value of 1.86V……..

Step 4:
Open the RG58 model file with the text editor and change this „R“-value. Repeat the simulation and the output voltage’s peak value is now at 1.86V.

Compare your simulation with that on the next page.



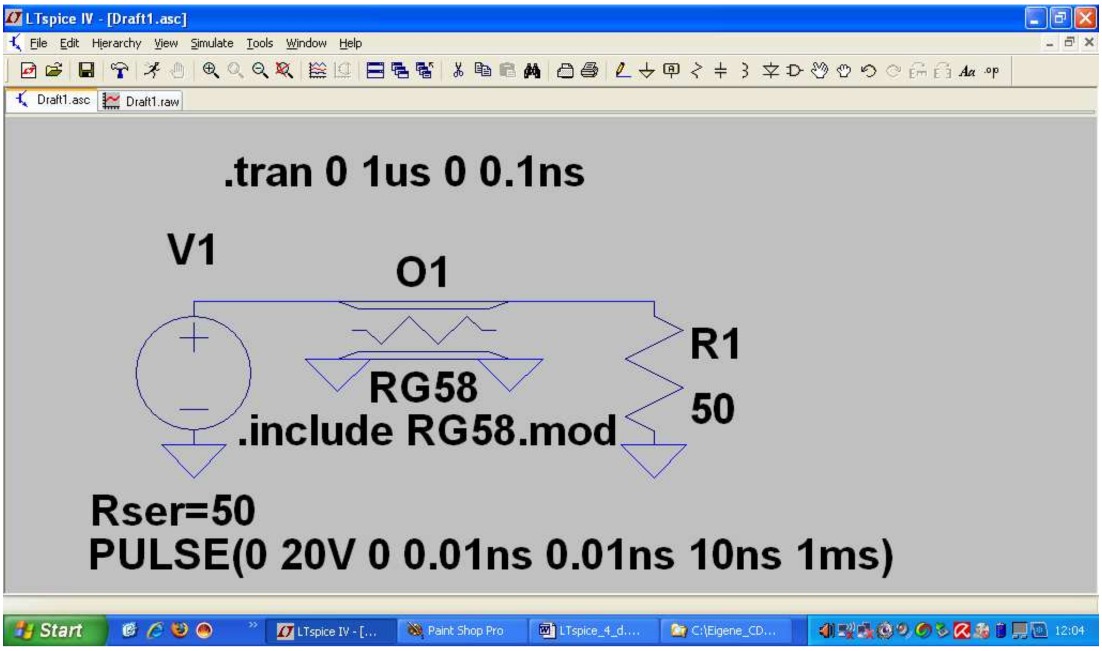
Peak Value = 1.86V

13.5.3. Feeding the RG58-Cables's Entry with a Pulse Voltage

We now test our „lossy line" with a pulse voltage at the input. Let the source properties be:

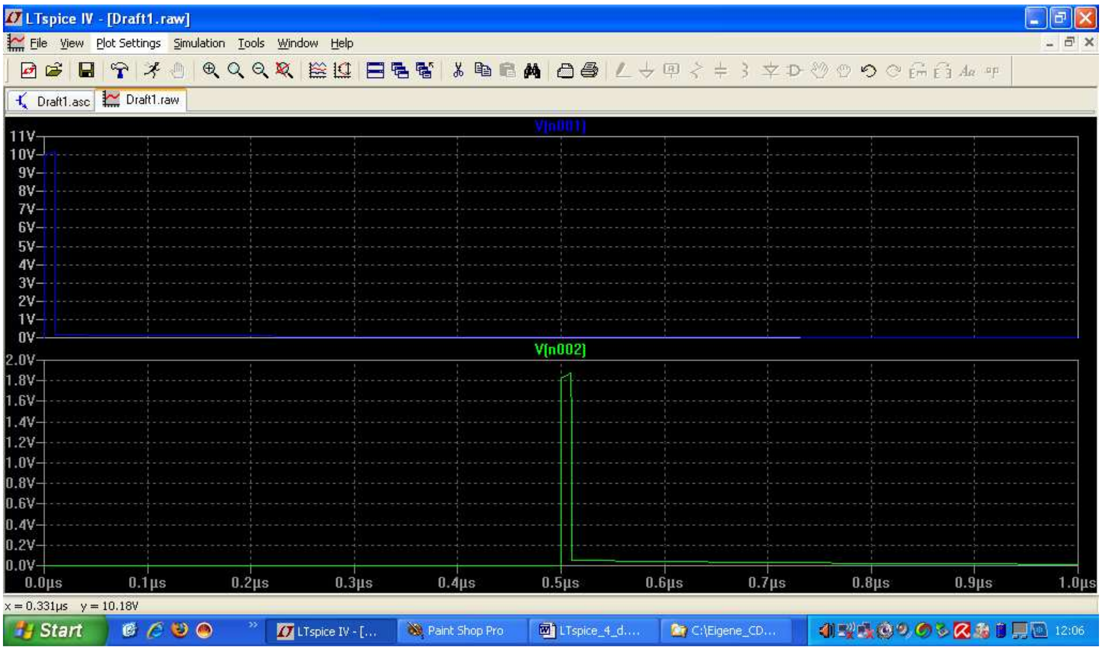
Source Voltage Maximum Value	**= 20V**
Source voltage Minimum Value	**= 0V**
Pulse Length	**= 10ns**
Rise Time	**= 0,01ns**
Fall Time	**= 0,01ns**
Periode Time	**= 1ms**

Schematic:



```
.tran 0 1us 0 0.1ns
.include RG58.mod
Rser=50
PULSE(0 20V 0 0.01ns 0.01ns 10ns 1ms)
```

Simulation Result:

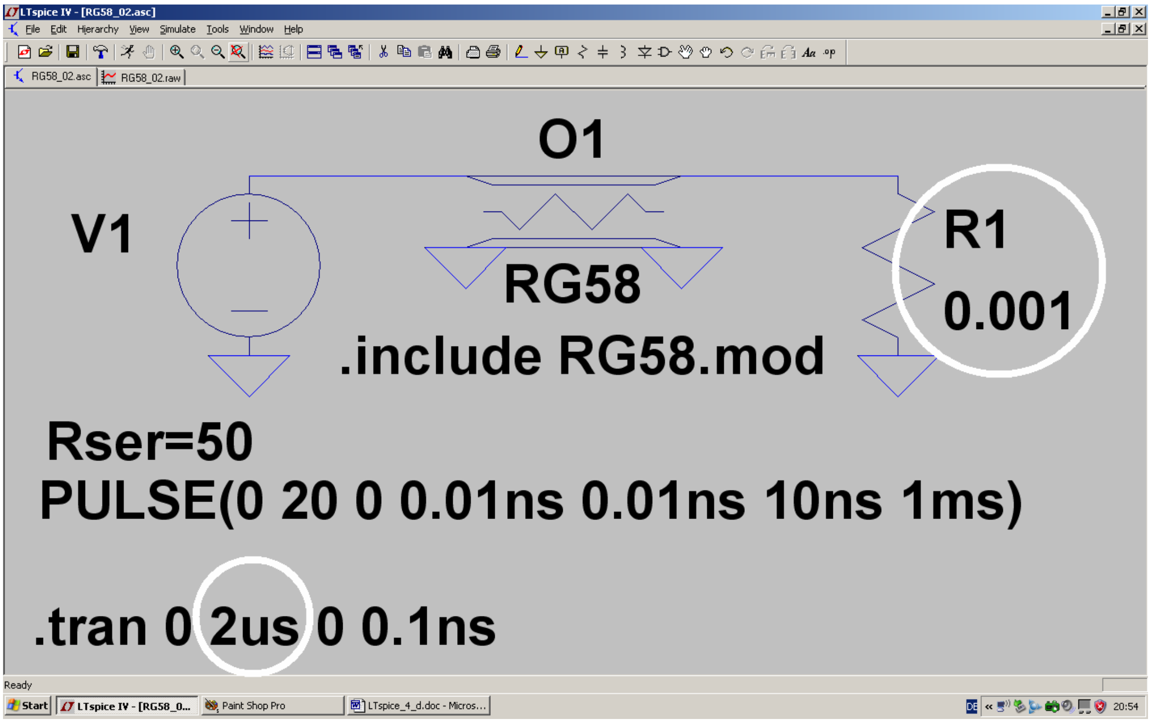


Because the value „R = 1.7Ω / 1m" does not change with frequency in the simulation (in comparison with the reality: really a pity!), all harmonics are attenuated by the same factor. So the curve form does not change as in practice (= no linear distortion). Only the amplitude at the cable's output is reduced.

13.5.4. A Short Circuit at RG58 Cable’s End

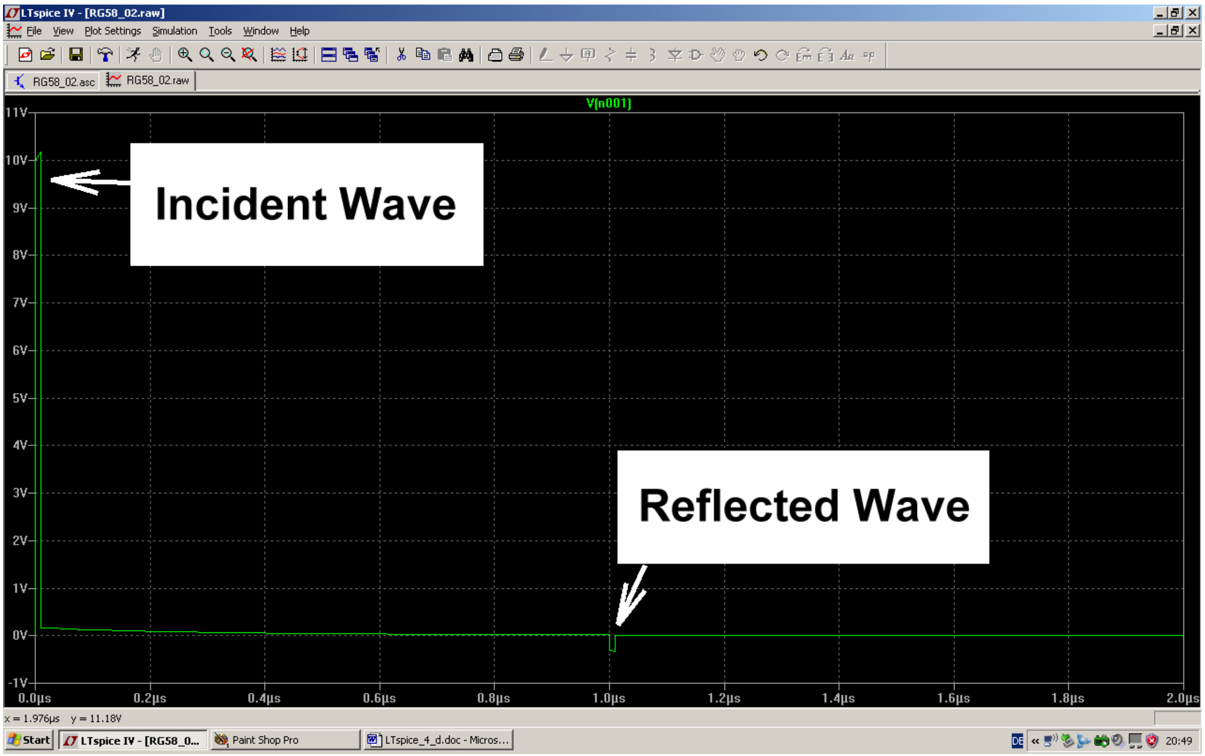
Do you remember? This is made using a termination of 0.001Ω, but we must not forget to increase the simulation time to 2ms (otherwise the echo cannot be seen in the waveform viewer).

You need not show the simulated output voltage in a (separate) diagram, because the voltage on a short circuit is “0”….so only the input voltage is of interest.



LTspice IV - [RG58_02.asc]
File Edit Hierarchy View Simulate Tools Window Help
RG58_02.asc
RG58_02.raw
O1
V1
RG58
R1
0.001
Ready
Start
LTspice IV - [RG58_0...
Paint Shop Pro
LTspice_4_d.doc - Micros...
DE
20:54

```
.include RG58.mod
Rser=50
PULSE(0 20 0 0.01ns 0.01ns 10ns 1ms)
.tran 0 2us 0 0.1ns
```



x = 1.976µs y = 11.18V

The echo is very small because of the huge cable attenuation.

14. Project 10: S-Parameters

14.1. Echos once again, but with more System (= S-Parameters)

Energy is always transported on a cable by travelling power. But for easier understanding and calculating we use the square root of the power to get an expression only with **voltage** and we name this result "**wave**".

So we get the „**incident wave a**": - $\mathbf{a} = \sqrt{\mathbf{P}_{\mathbf{incident}}} = \sqrt{\frac{\mathbf{U}_{\mathbf{incident}}{}^{2}}{\mathbf{Z}}} = \frac{\mathbf{U}_{\mathbf{incident}}}{\sqrt{\mathbf{Z}}}$

And the „**reflected wave b**": $\mathbf{b} = \sqrt{\mathbf{P}_{\mathbf{reflected}}} = \sqrt{\frac{\mathbf{U}_{\mathbf{reflected}}{}^{2}}{\mathbf{Z}}} = \frac{\mathbf{U}_{\mathbf{reflected}}}{\sqrt{\mathbf{Z}}}$

The connections of a part are now named "ports". A simple block (= filter, amplifier, attenuator...) with input and output is then a **TWO PORT**, but a resistor or a diode is a **ONE-PORT.**

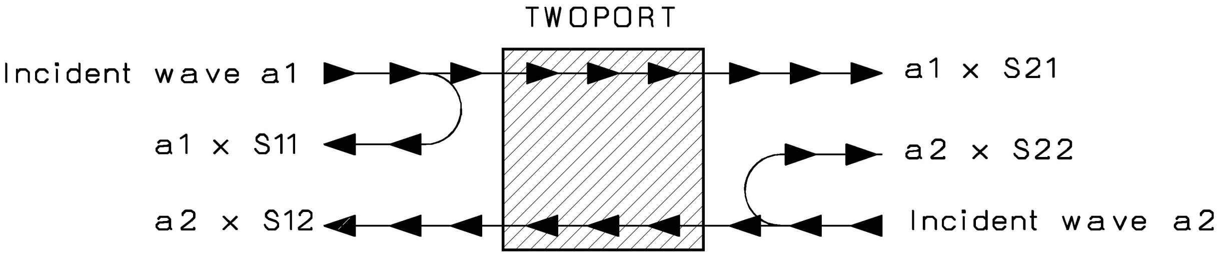
.

TWOPORTs have additionally a gain or an attenuation. This means:

The incident wave at a port will therefore produce a signal at the other port.

This effect can be measured and expressed as „transmission".

If we look at an amplifier as a typical TWOPORT, then we get the following **signal flow chart**:



Input side = port 1:
The incident wave a1 (coming from the signal source) feeds the input (port 1) of the TWOPORT. In general we find then a wave „b1" running back in direction of the signal generator. This "**backrunnning wave b1**" can be separated from the incident wave a1 by a directional coupler and consists of two parts:

1) **The reflected wave „a1 x S11"** due to the fact that the input impedance at port 1 isn't exactly 50Ω and

2) a **second signal** which is normally caused by „**feedback"** (= reverse transmission) from the output to the input.

So the total echo b1 on Port 1 can be expressed as $b_1 = a_1 \bullet S_{11} + a_2 \bullet S_{12}$

If the output (port 2) is terminated with exactly Z = 50Ω, then no echos come back from the termination resistance in direction of port 2. Now the incident wave a2 at the output is zero and we get a simple solution for S11:

$$S11 = \frac{b_1}{a_1}$$ for a_2 = zero

But…this is the well – known **Input Reflection Coefficient "r"** of the preceeding example! In most cases „r" is complex and so the part manufacturers always list MAGNITUDE = MAG and ANGLE = ANG in their S-parameter-files for different frequencies.

S12 is then the „reverse transmission coefficient“. In this case we terminate the input (Port 1) by 50Ω and disconnect the input signal generator. The signal generator is now connected to the output (port 2) to measure the feedback effects at the input (port 1).

$$S_{12} = \frac{b_1}{a_2} \qquad \text{for } a_1 = \text{zero}$$

Output side = port 2:
One signal generator is connected to the output (port 2) and sends the wave a2 to this port. The other signal generator feeds the Input of the Twoport.

With a directional coupler we can at the output separate the incident wave a2 and the “echo b2” coming back from the TWOPORTs output:

$$b_2 = a_1 \bullet S_{21} + a_2 \bullet S_{22}$$

If we now disconnect the signal generator from the input and terminate with 50Ω, then the output’s echo consists only of the „reflected wave “**a_2 x S22**” (if the output impedance differs from 50Ω). So S22 is simply the „output reflection coefficient for perfect matching at the input (= port 1)”.

$$S_{22} = \frac{b_2}{a_2} \qquad \text{for } a_1 = \text{zero.}$$

Now let us look at the rest of the formula:
If the input signal generator is switched ON and the output signal generator disconnected and replaced by a perfect 50Ω-Match, we get only the wave

$$\mathbf{b_2 = a_1 \times S21}$$

at the output. So S21 = forward transmission coefficient means simply the gain or attenuation of a TWOPORT (with perfect match at the output)!

$$S_{21} = \frac{b_2}{a_1} \qquad \text{for } a_2 = \text{zero.}$$

Please remember:
The S-Parameters are **„wave“ ratios** -- that means: voltage ratios. So if you want to calculate the power gain, you must use **$(S21)^2$**

How can we get an input or output impedance value from S11 resp. S22?

Use the following formulae, but remember: normally all the numbers are complex……:

Input resistance: $$\mathbf{Z_{Input} = Z \bullet \frac{1+S11}{1-S11}}$$

Output resistance: $$\mathbf{Z_{Output} = Z \bullet \frac{1+S22}{1-S22}}$$

It is also possible to calculate the reflection-S-parameters from measured impedance values, but therefore you need a modern microwave CAD-program (like the free ANSOFT DESIGNER SV) to save lot of efforts and sweat.

And lastly let us have a look at the S-Parameter-file of a modern MMIC (= “Touchstone” or “S2P” file), downloaded from the Internet:

```
!.....INA-03184....S-Parameters
!       Id = 10 mA       LAST UPDATED 07-22-92
# ghz S ma r 50
0.0     .32     180     19.2     0      .014     0      .55      0
0.05    .32     179     19.14   -3      .014     3      .55      0
0.10    .32     176     19.05   -7      .014     4      .57     -3
0.20    .32     172     19.05   -14     .014     6      .55     -5
0.40    .32     165     18.78   -29     .014    10      .53    -11
0.60    .32     158     18.71   -43     .015    11      .51    -14
0.80    .32     151     18.53   -57     .015    13      .51    -17
1.00    .32     144     18.18   -72     .016    21      .50    -20
1.20    .30     135     18.27   -86     .016    25      .50    -23
1.40    .31     126     18.10   -102    .017    30      .49    -29
1.60    .30     117     17.92   -117    .018    38      .48    -34
1.80    .26     102     17.49   -135    .019    44      .45    -41
2.00    .22      92     16.62   -153    .020    49      .40    -50
2.50    .09      91     12.88    168    .021    57      .26    -48
3.00    .14     160      8.79    134    .023    65      .22    -33
3.50    .24     151      5.92    108    .025    69      .26    -33
4.00    .29     139      4.18    87     .029    81      .28    -43
```

And this is how to interprete these files:

a) A line starting with an exclamation mark is a comment and therefore ignored for calculation

b) The line **# ghz S ma r 50“** means:

The first value is the **measuring frequency in GHz.**

Then follow the 4 S-parameters in the range **S11 S21 S12 S22**

“ma” says: every S-parameter is given as its **magnitude**, followed by the **angle**.

“r 50” tells us that the system resistance (for measurement and application) is real 50 Ohm.

Example line for **f = 1GHz**:

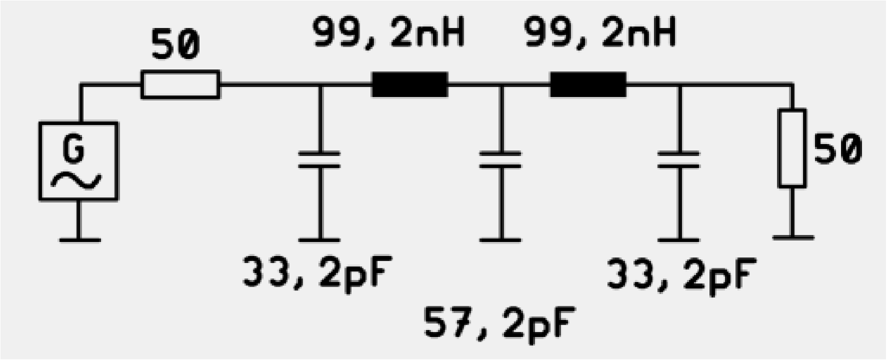
S11 = 0.32 / 144 degrees	(Magnitude of the input echo is 0.32 x a1, the phase of the echo is 144 degrees -- caused by an input impedance **not equal to 50Ω**).
S21 = 18.18 / -72 degrees	**When the output (= Port 2) is terminated by 50Ω, then we get a voltage gain of 18.18 (= ratio of output wave a2 referred to input incident wave a1)**
S12 = 0,016 / 21 degrees	The feedback voltage measured at the input (= port 1) is 0.016 =1.6% of the output wave a2
S22 = 0.5 / -20 degrees	Magnitude of the output echo is 0.5 x a2, phase of the echo is 20 degrees -- caused by an output Impedance **not equal to 50Ω**

14.2. Example: 110MHz – Tchebyshev Lowpass Filter (LPF)

Let us now simulate the S-parameters of a 110MHz Tchebyshev lowpass filter. This type of filter suffers from “passband ripple” but provides a sharper transition from the passband to the stopband.

Properties:

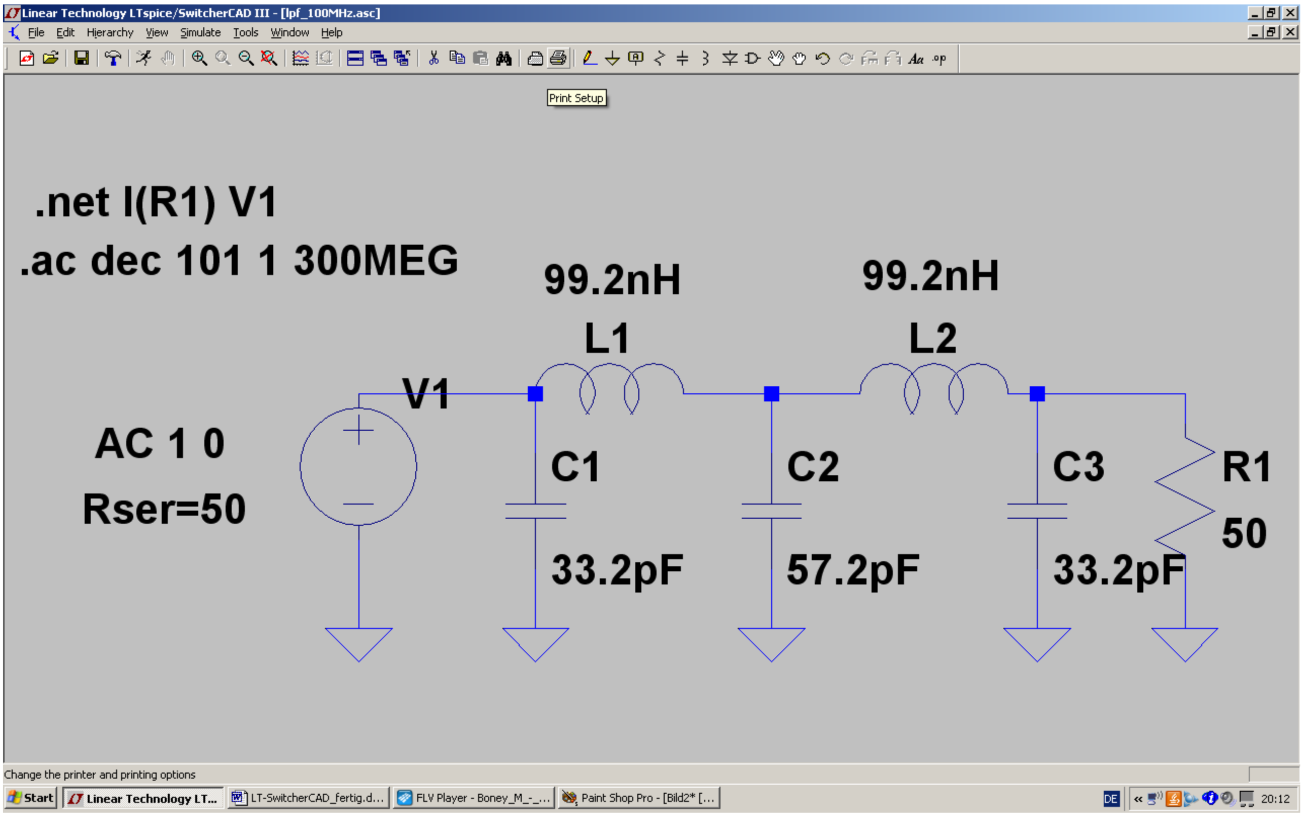
„Ripple“ corner frequency	fc = 110 MHz
Filter degree	n = 5
System resistance	Z = 50Ω
„Passband ripple“	0.1 dB.
(This ripple causes a maximum value of S11 = -16,4 dB)	



For the filter part calculations use the free microwave CAD software (Ansoft Designer SV) with the integrated filter designer and you get:

C1 = C3 =	**33.2 pF**
C2 =	**57.2 pF**
L1 = L2 =	**99.2 nH**

And this is the simulation schematic for LTspice:


Linear Technology LTspice/SwitcherCAD III - [lpf_100MHz.asc]
File Edit Hierarchy View Simulate Tools Window Help
Print Setup
99.2nH
L1
99.2nH
L2
V1
AC 1 0
Rser=50
C1
33.2pF
C2
57.2pF
C3
33.2pF
R1
50
Change the printer and printing options
Start
Linear Technology LT...
LT-SwitcherCAD_fertig.d...
FLV Player - Boney_M_-_...
Paint Shop Pro - [Bild2* [...
DE
20:12

```
.net I(R1) V1
.ac dec 101 1 300MEG
```

Explanations:

a) The directive **.net I(R1) V1** starts a network calculation with the output voltage referred to the input voltage (given by source V1).

b) Then program a decadic sweep from 1Hz to 300 MHz with 101 points per decade by the directive

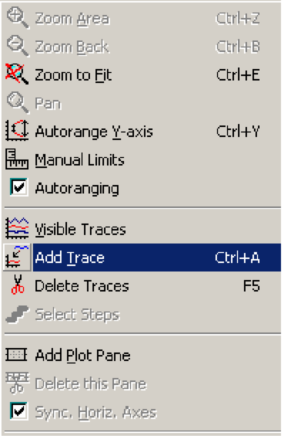
.ac dec 101 1 300MEG

c) For a sweep the properties of source V1 must be set to „AC amplitude =1 and AC phase = 0“:

AC 1 0.

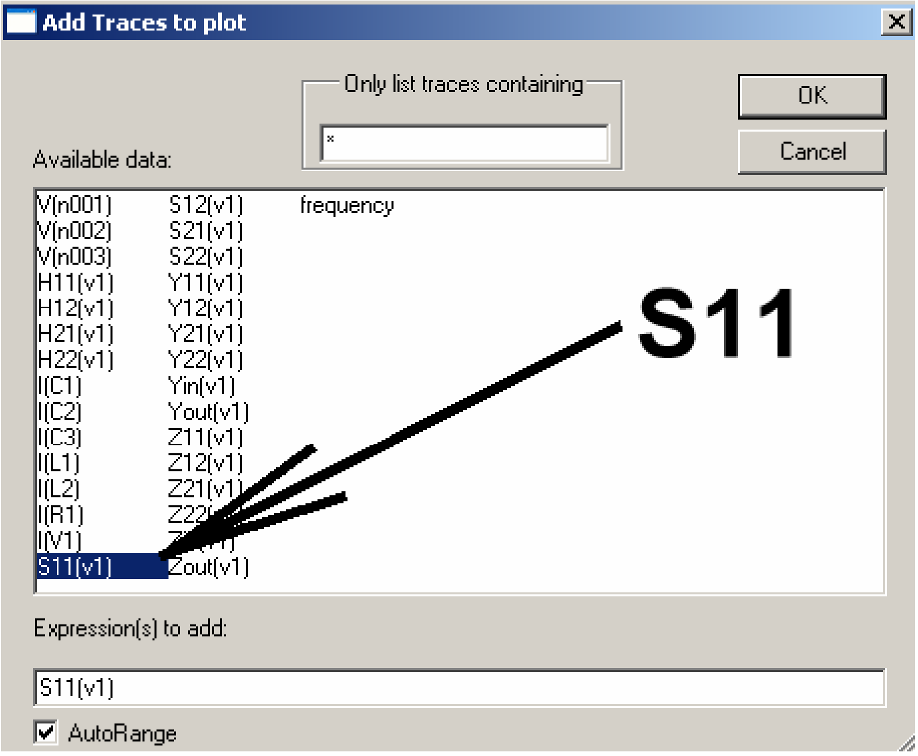
d) After a right mouseclick on the symbol of source V1 enter the value for the necessary source resistance of 50Ω in the system:

(Rser=50)



Now press the simulation button and follow these steps:

Step 1:
The diagram for the results is still empty. So click right on it with your mouse and choose **„Add Trace“**



Expression(s) to add:

S11(v1)

☑ AutoRange

Step 2:
Find

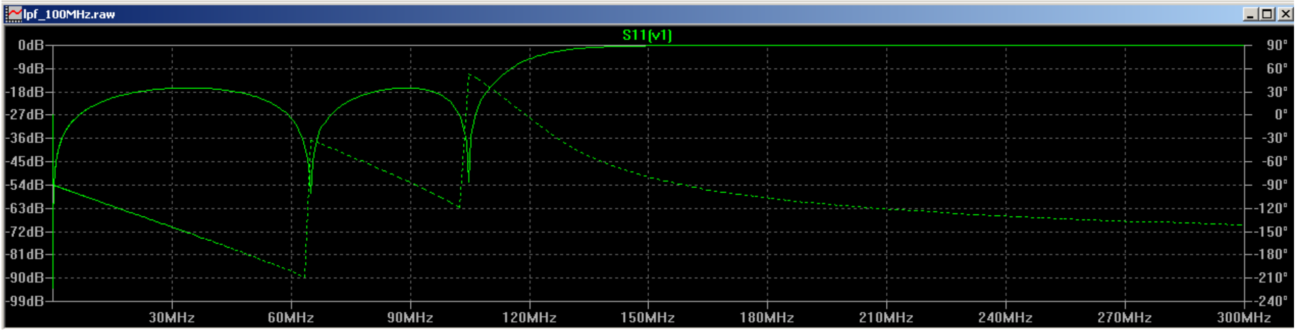
„S11(v1)“

in the list, mark it and

press OK.

For professionals:
In this list also the Y- and Z-parameters can be found...

Step 3:

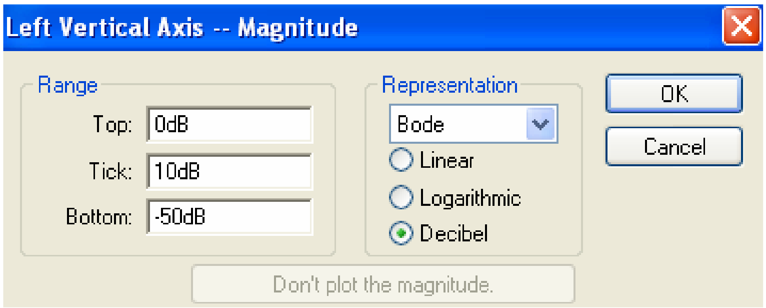


The result is not so bad, but the presentation…….
The phase curve might be interesting, but we want to delete it. The dB-scaling of the vertical axis must be improved.

So move the cursor to the scaling of the **right vertical axis** and click on it. Choose in the menu

Don't plot phase.

The phase curve disappears at once.



Left Vertical Axis -- Magnitude

Range

Top: 0dB

Tick: 10dB

Bottom: -50dB

Representation

Bode

○ Linear

○ Logarithmic

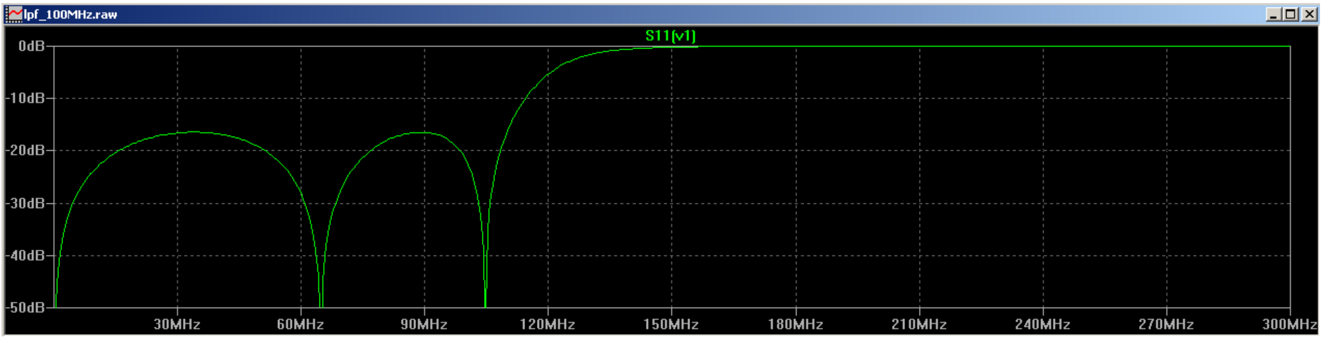
● Decibel

Now we repeat this procedure with the left vertical axis and enter the values of the left figure (= range from 0dB to -50dB with a tick of 10dB).

At last and once more click right on the diagram to activate the option

GRID

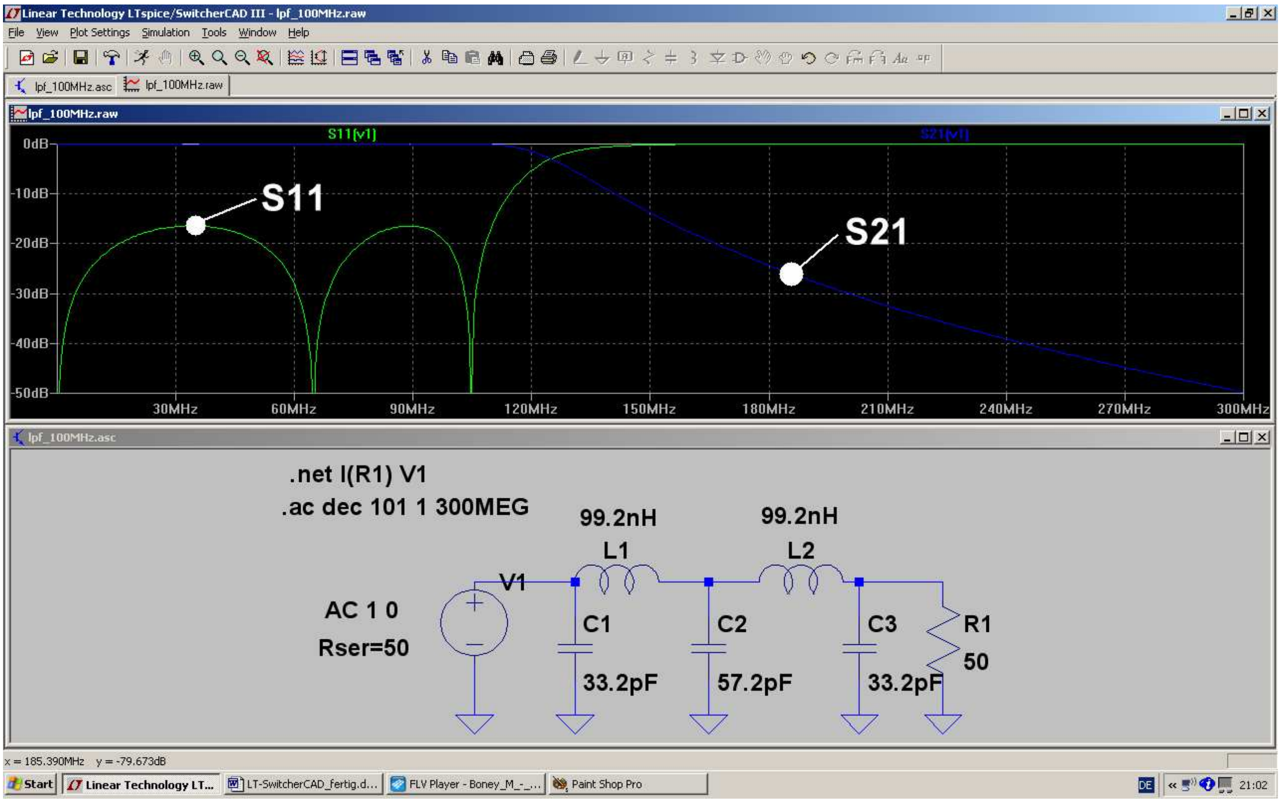
Now you have this result:



Step 4:
We complete the work by adding the S21 curve. Let's go:

Right mouseclick on the diagram / Add Trace / choose S21 and press OK.

Sorry, but it is now better to repeat the procedure with the scaling of the left vertical axis to show a range from 0....-50dB. This gives the final result on the next page.



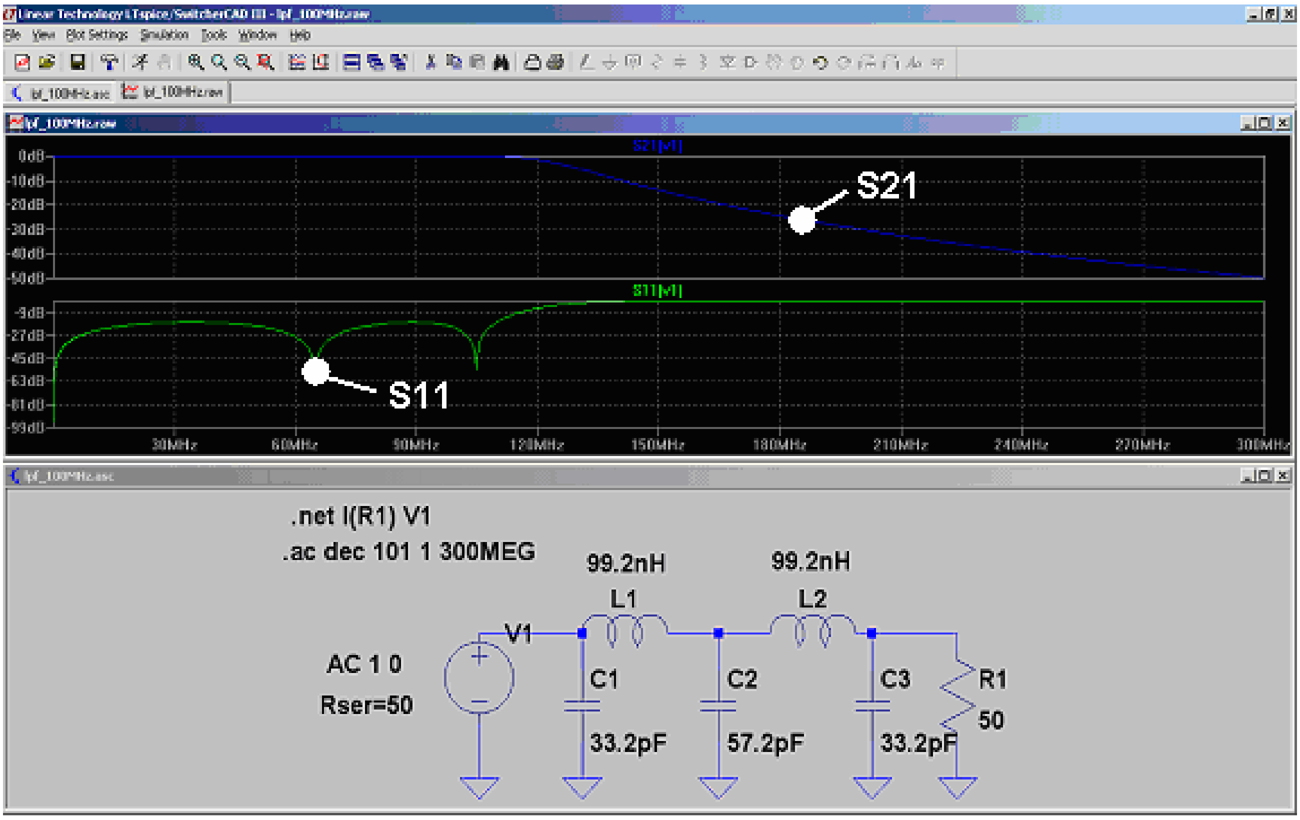
```
.net I(R1) V1
.ac dec 101 1 300MEG
```

```
x = 185.390MHz   y = -79.673dB
```

For a separate presentation of the two S-parameters right click on the diagram and choose

Add Plot Plane

for the parameter S21. Delete the S21-curve in the „old“ diagram and scale both diagrams again:



```
.net I(R1) V1
.ac dec 101 1 300MEG
```

15. Project 11: Double Balanced Mixer (= DBM = Ring Modulator)

15.1. Fundamentals and Definitions

This well known circuit is used for converting frequencies without changing the information contents.

We have two inputs and one output:

a) RF = radio frequency input. Here the applied signal amplitude must be small (typically below 50...100mV)

b) LO = local oscillator input. Here the signal must be high (typically up to several volts)

c) IF = intermediate frequency output. Here we find the converted signals.

Note:

This operates a multiplier circuit for the two input signals!

When two sine signals are multiplied we get the following result:

$$\sin(\alpha) \bullet \sin(\beta) = \frac{1}{2}[\sin(\alpha+\beta)+\sin(\alpha-\beta)]$$

This means for our circuit:

At the output of the DBM the two input signals have disappeared and are now replaced by their sum frequency and difference frequency!

When using a square wave signal as the LO-input, then this signal consists of the fundamental frequency f1 and the harmonics (with 3 x f1 , 5 x f1 , 7 x f1 , 9 x f1......).

Not only the fundamental frequency but also the harmonics are now multiplied by the RF-signal in the DBM and produce sum and difference frequencies with amplitudes that decrease linearly with the “harmonic’s degree”.

So you get at the DBM’s output a lot of „signal pairs“ (= sum- and difference frequencies). Every pair is located at a harmonic frequency.

In practical applications the desired new frequencies must now be picked out with a lowpass or a bandpass filter by the user.

15.2. The Ring Modulator

This part is used everywhere in communication systems and is offered by a lot of manufacturers -- as an active or passive device. We'll examine the standard passive version.

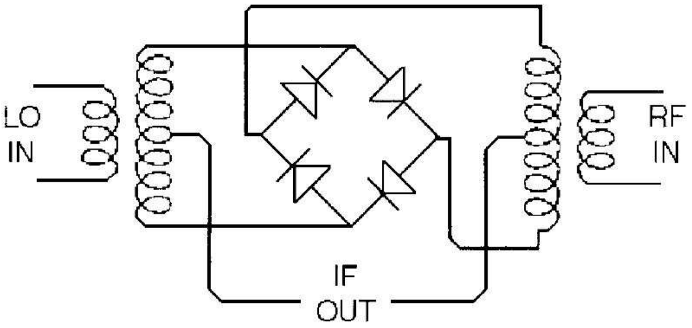


Figure 2. A popular diode-ring mixer topology.

A ring of four schottky diodes is connected to two transformers. Each transformer uses 3 identical windings. The two secondary windings are connected in series.

The use of superfast schottky switching diodes and special microstrip circuits enables cutoff frequencies higher than 50....100GHz in special applications.

Explanation:

„**LO-IN**" means „**local oscillator input**" for the converting signal.

„**RF-IN**" means „**radio frequency input**" for the information which shall be converted.

„**IF-OUT**" means „**intermediate frequency output**" and delivers the multiplication result.

Principle of operation:

The LO-signal at the secondary winding of the left transformer (amplitude: up to several volts….) switches the right diode pair ON at the positive halfwave and the left diode pair ON at the negative halfwave.So either the upper secondary winding or the lower secondary winding of the right transformer with the RF signal is connected to the IF output by the conducting diode pair. But the RF signal which finally reaches IF-OUT is changing its polarity rhythmic with the LO frequency. That is the same effect as multiplying the RF – sinewave with a LO-squarewave (amplitude values: +1 and -1).

Advantages: Very low distortion. Useful for broadband applications up to many GHz. Needs no power supply, but correct 50Ω-matching at all ports. Produced in huge quantities at reasonable prices by a lot of manufacturers.

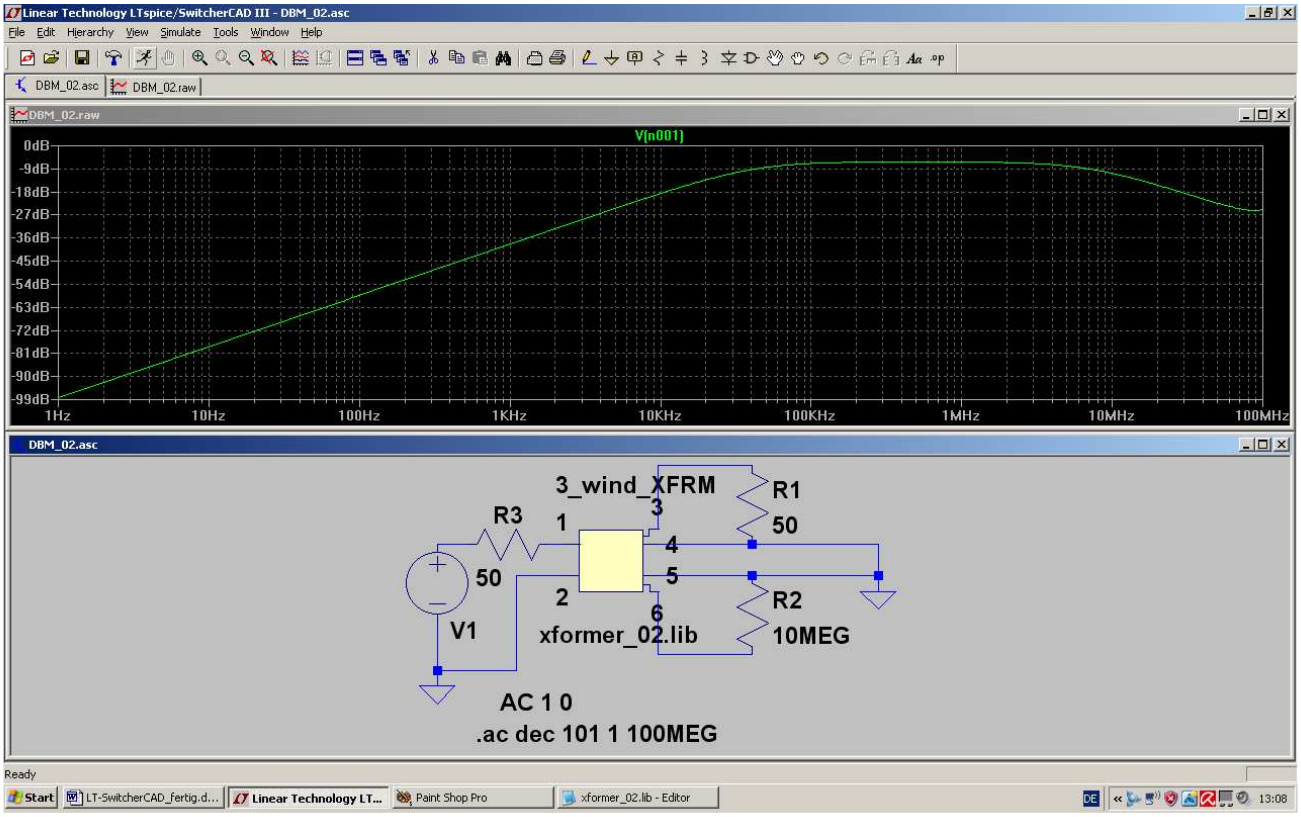
Disadvantages: Due to the missing power supply the complete „switching energy" must be delivered by the LO signal source. It demands high LO levels for rapid diode-switching and low distortion. There is no gain but always an attenuation of 5….8dB between the RF input and the IF output. This increases the noise figure in a system and the attenuation must be compensated by more „pre-amplifying".

15.3. The necessary Transformers

The „weak points“ of this circuit are always the 2 broadband transformers …and the special construction is a top secret guarded by the manufacturers! The transformers will in any case limit the bandwidth and the upper frequency.

So let us study an example with LO frequency = 1MHz and RF frequency = 100kHz. The lower cutoff frequency of the transformer should therefore be below 100kHz and the upper frequency limit min.10….20 MHz.

We use our well known „xformer_02“ with 3 identical windings and vary the properties to reach these specifications.
But remember: when the mixer is operating, always only one of the two secondary windings is in action, while the other is “idling” as long as the active diode pair is switched ON! So in the transformer simulation only one secondary winding is terminated by 50Ω while the other winding is simply feeding a 10MΩ-Resistor. This avoids the error message “node is hanging free”.
The main winding inductance is 100µH, copper resistance is 1Ω and winding capacitance is 1pF:



```
AC 1 0
.ac dec 101 1 100MEG
```

Now let’s tackle the complete mixer!

15.4. DBM Simulation with ideal Transformers

From the left comes the LO signal to switch ON and OFF the Schottky diodes (sine wave, peak value = 2V, frequency = 1MHz). From the right comes the information as RF signal (sine wave, peak value = 20mV, frequency = 100kHz.

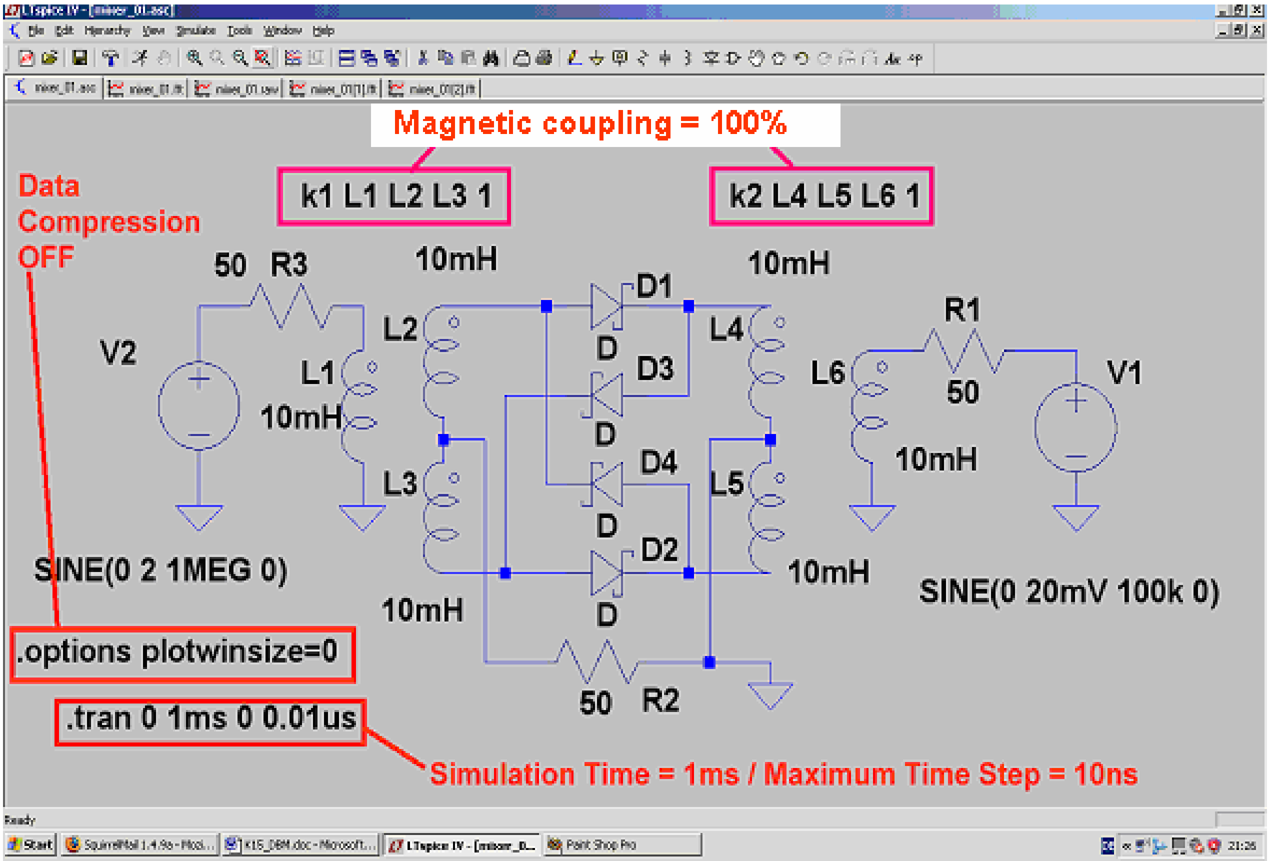
Both ideal transformers use 3 separate windings(each L= 10mH) which are magnetically coupled. The coupling factor is 1 (= 100%).

The Schottky diode comes from the LTspice part library. The symbol is named “schottky”.

A simulation time of 1ms gives a frequency resolution of 1/1ms = 1000Hz. So for a number of 100 000 true samples a maximum time step of 1ms / 100 000 = 10ns must be used. This time step corresponds to a minimum sample frequency of 1/10ns = 100MHz and so no aliasing effect should be found up to a signal frequency of 50MHz.

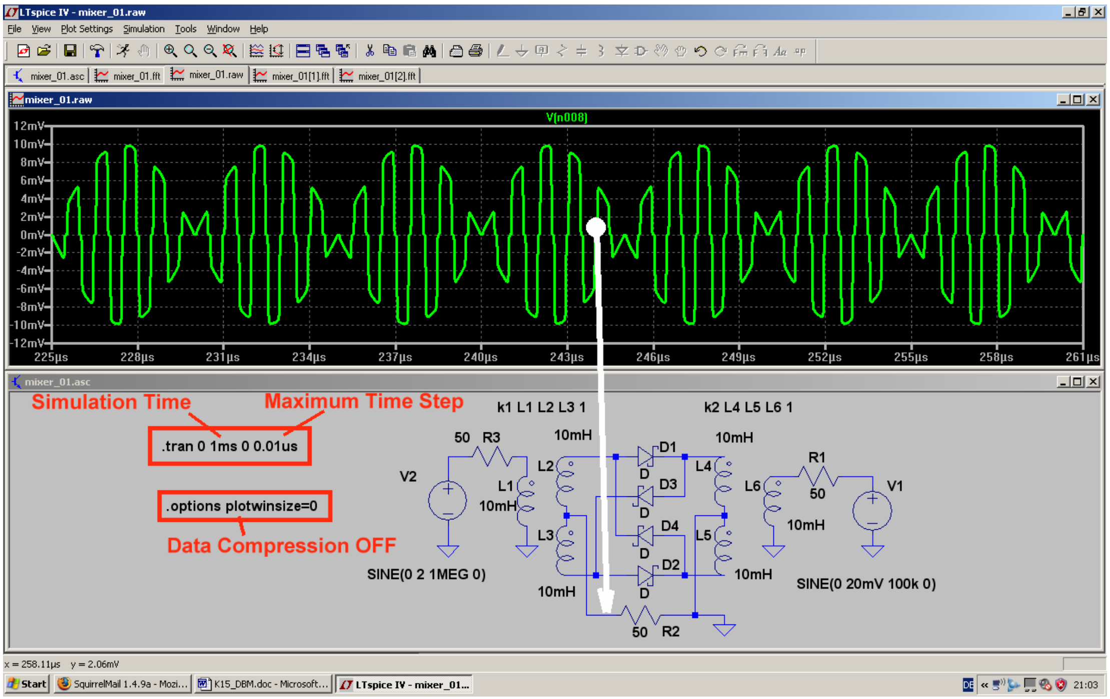
Data compression is switched OFF by the spice directive **.options plotwinsize=0**.

This is the screen before starting the simulation:



```
k1 L1 L2 L3 1
k2 L4 L5 L6 1
.options plotwinsize=0
.tran 0 1ms 0 0.01us
```

And so the screen looks like this after the simulation of the output voltage at resistor R2 and zooming



Simulation Time

Maximum Time Step

.tran 0 1ms 0 0.01us

.options plotwinsize=0

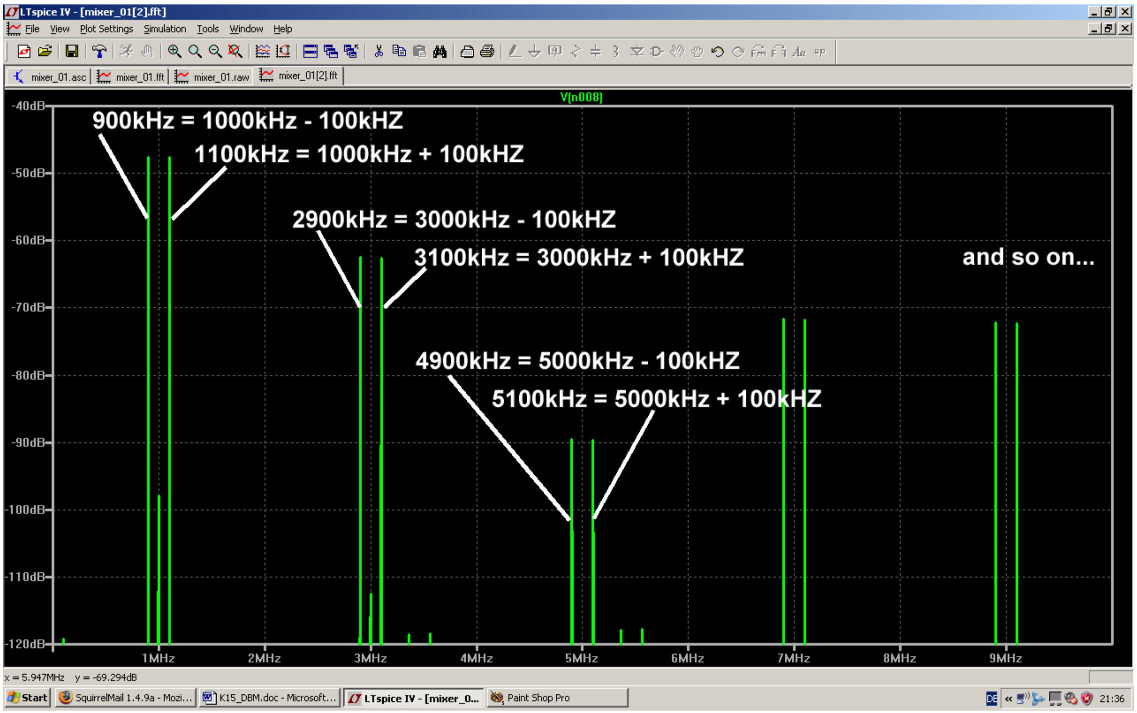
Data Compression OFF

The polarity changing of the RF signal (caused by the LO signal) can clearly be seen. But: the „ON“ voltage of 0.3V of every Schottky diode causes distortion at every zero crossing.

Lastly let us look at the FFT result for the output voltage. So right click on the time domain curve, open “View” and “FFT” and use **65536 samples** in time.

Press OK and modify the scaling of the diagram axis:

Frequency axis: **linear ; start frequency = 0Hz ; stop frequency = 10MHz ; tick = 1MHz**
Vertical axis: **dB scaling from -40dB .....-120dB with a tick of 10dB.**



900kHz = 1000kHz - 100kHZ

1100kHz = 1000kHz + 100kHZ

2900kHz = 3000kHz - 100kHZ

3100kHz = 3000kHz + 100kHZ

and so on...

4900kHz = 5000kHz - 100kHZ

5100kHz = 5000kHz + 100kHZ

Here comes the confirmation of theory:
At every LO harmonic frequency (3MHz ; 5MHz ; 7MHz ; 9 MHz…) we find a signal pair consisting of the sum and the difference frequeny of the RF signal and the harmonic.

Neither the RF signal nor the LO signal nor the LO harmonics should be found at the output.

16. Project 12: Simulations with Digital Circuits

16. 1. What you should know before you begin

Sorry, but this information is not described in the online help or the included examples. So you need to gain some of your own experience (..also gained through the use of other SPICE-programs) by exploring the problems and to master these non-intuitive constraints .

The most important rules for the „normal user“ are:

The included library **„[Digital]“** contains only **„ideal fundamental blocks“.**

Each of them always has 8 pins!

The reason for this is:

a) You always have several input pins (e.g. an AND-Gate has 5). Please leave all non-used pins open-circuit (unconnected). They will be ignored for the simulation.

b) Normally you will get **not only the desired output signal** but also the **inverted signal**.

c) Logical levels are „**0 Volt“ for „Logical Zero” and „+1 Volt“ for „Logical 1“**. The internal threshold voltage is 0.5V.

If you need to work with other voltage levels (like TTL = 0V /+5V), then right click on the part symbol, open the line „value“ and write the following entry:

Vhigh=5V Vlow=0V

d) **Output pins should not be hanging free in the air.** So please **add a label** or terminate them by a 10kΩ – resistor to ground for a troublefree simulation.

e) The supply voltage pins are completely missing….you are working with “ideal parts“.

Use the „PULSE-“ or the PWL voltage as input signal.
Minimum value is 0 Volt, maximum Value is = +1Volt (…other values are handled as discussed above…)
As rise and fall time use 1ns.

To see your simulation results open separate diagrams by right clicking on the result screen and choosing

Add Plot Plane

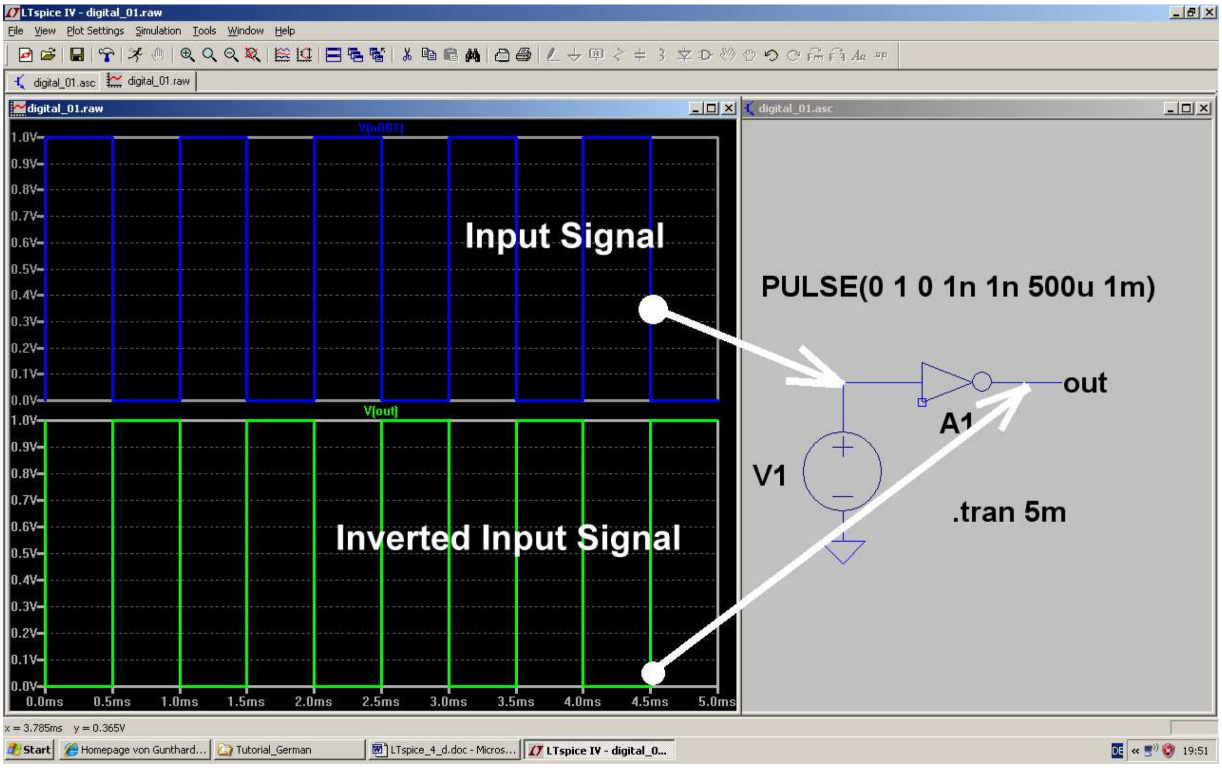
Use as many separate diagrams as you need to present signals -- otherwise you go crazy after a short time. Filling such a plot pane is very simple: click right on the diagram and choose “Add Trace”.

16.2. Simple start: the Inverter (= NOT)

Get the part as „**Digital / INV“** from the part library and use as the input signal a symmetric pulse voltage (Umin = 0V / Umax = +1V / Rise Time = Fall Time = 1ns / Pulse Length = 500µs / Period Time = 1ms). Simulate for 5ms.

Connect the Voltage Source to the upper input pin and leave the other input pin open-circuit.

Now simulate, open two diagrams by using „**Add Plot Plane**“ and view the input and the output voltage. You will get the following screen when selecting „**Tile vertically**“ in the „**Window**“ menu:

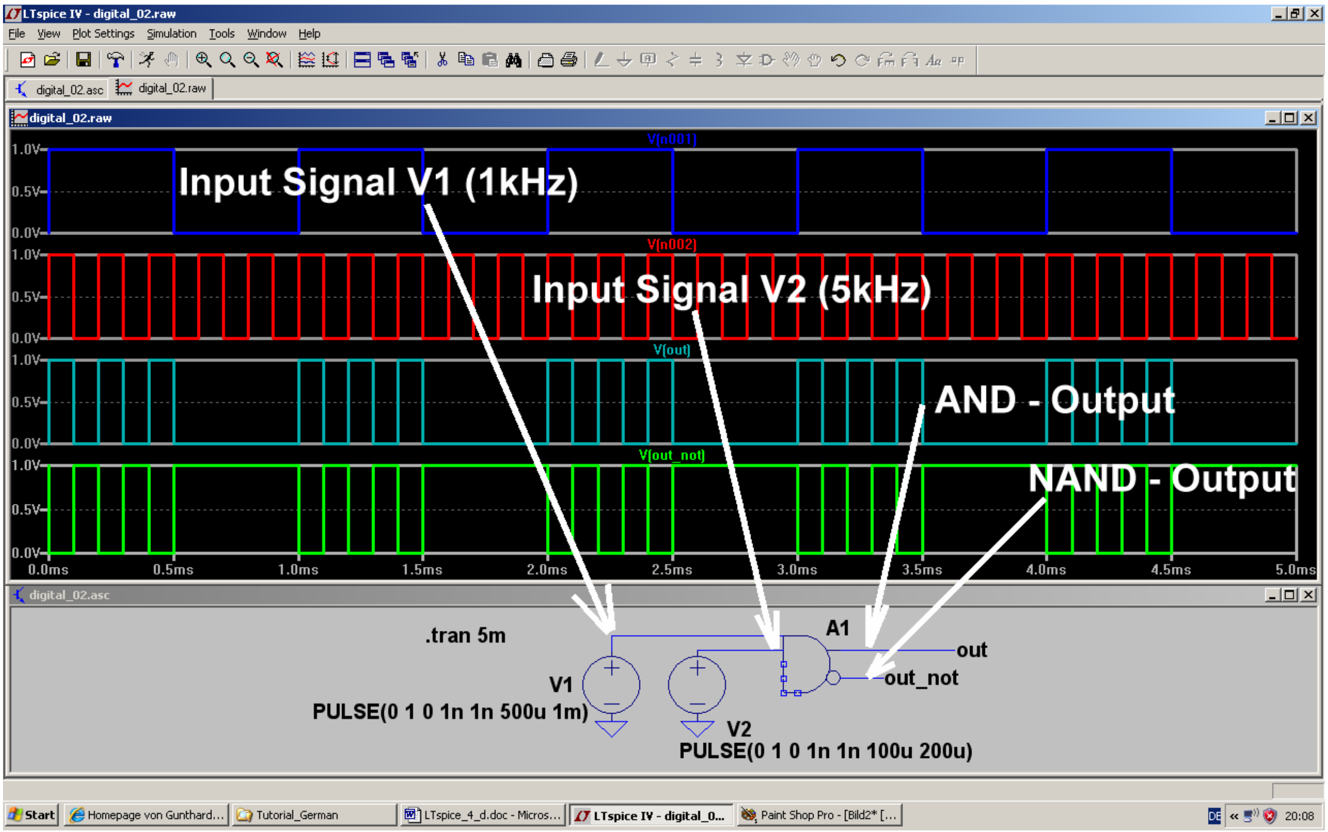


16.3. AND- and NAND Gate

Let us simulate an AND Gate with 2 inputs. One input is fed by a symmetric pulse signal of 1kHz, the other input with a symmetric pulse signal of 5kHz. Simulate for 5ms.

As already mentioned, leave all unused input pins open-circuit and use a label for the output pin.

Here is the simulation result:

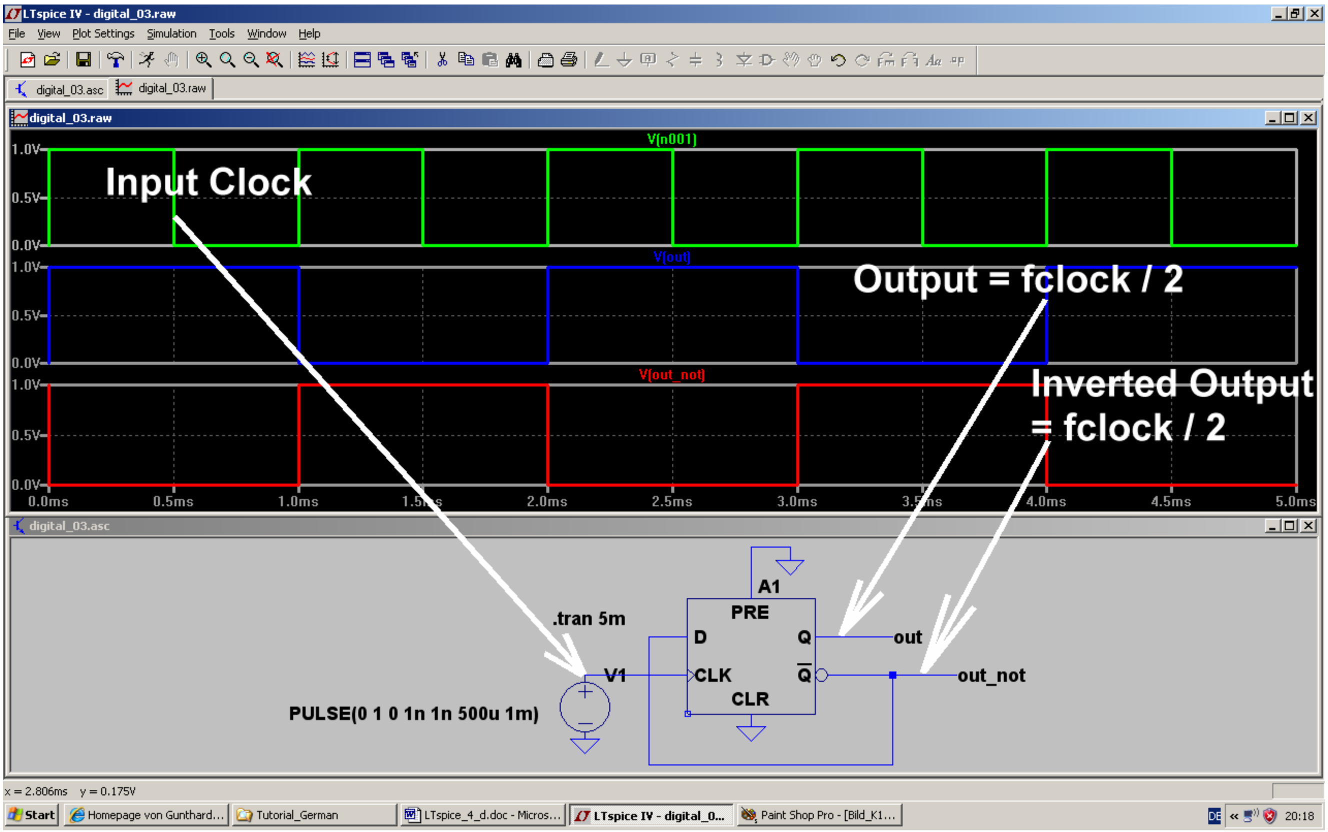


.tran 5m

PULSE(0 1 0 1n 1n 500u 1m)

PULSE(0 1 0 1n 1n 100u 200u)

16.4. D Flipflop

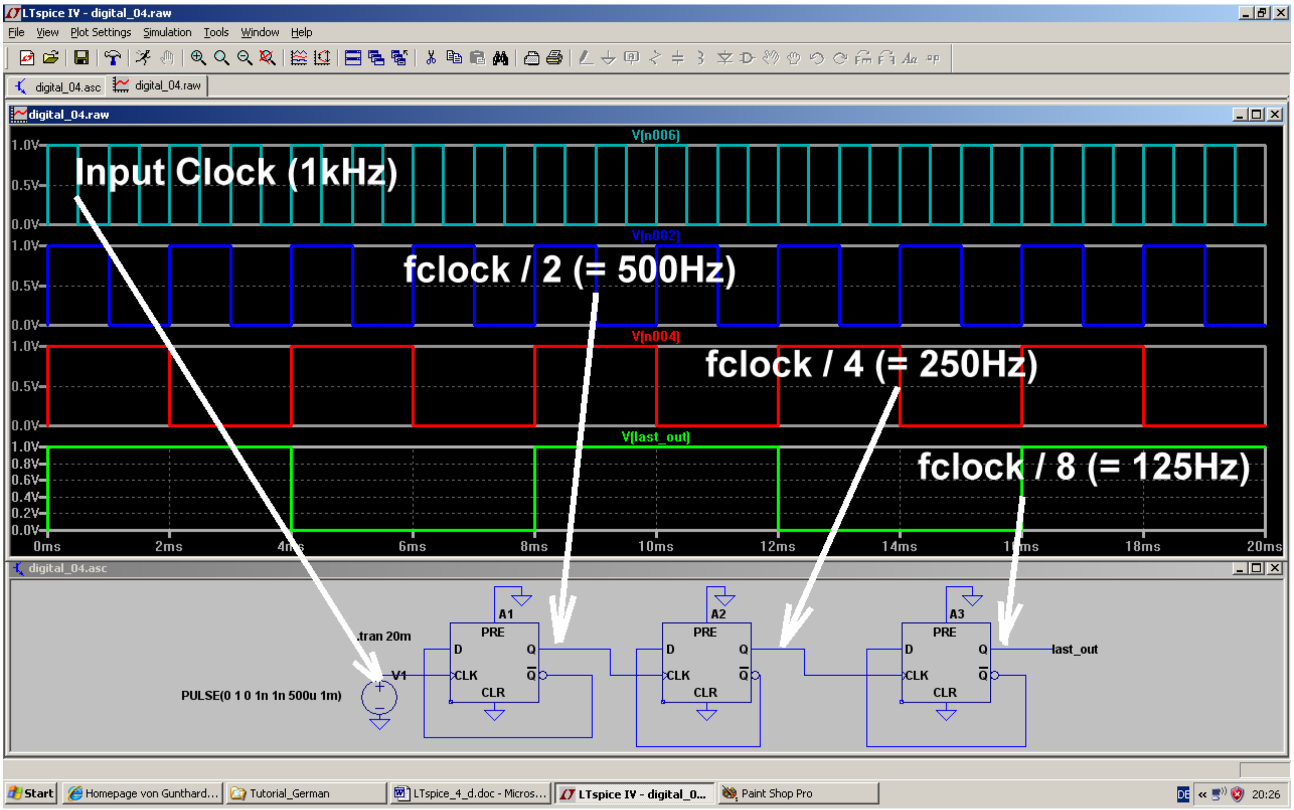


This is the well known „Binary Divide by Two Frequency Divider“. At the output you get one half of the input clock frequency. The inverted Q-output is connected to the D-input pin.

„Preset“ and „Clear“ must not be active and so they are connected to ground.

16.5. Three Stage Frequency Divider with D flipflops

This is a very simple task because you only have to connect three D-stages in series and wire them. Simulation time is 20ms, the input clock is a square wave of 1kHz.



If anyone has noticed that this chapter is short in comparison to the others: you are right. The reason is simple: in the LTSpice library the complete TTL and CMOS part collection is missing. This is really a pity when you consider that all these interesting circuits and schematics and problems could be simulated.

But there is a way -- this link:

http://tech.groups.yahoo.com/group/LTSpice/files/%20Lib/Digital%2074HCTxxx/

It’s the address of the LTSpice usergroup and there you find all the missing parts and symbols of the 74xx-families.
Thanks to Helmut Sennewald who has done the work of creating these important parts. Once more: Thanks, Helmut, for all your personal effort!

But:
This is a community and so you have to register first to get your ID and password. (no problem, it is free). If you do this you will be astonished and pleased: you’ll find extensive Online discussion consisting of calls for help, questions, answers, remarks, advice, offers, problem solutions....and a lot of nice and kind people, always ready and willing to help.

So why wait? Go and see.

17. Project 13: Noise Simulation

17.1. Fundamentals

Please note
The following 3 chapters are part of my article

„Practical Project: Noise factor measurement with older Spectrum Analysers, Part 1 and 2“.

Published in „VHF –Communications”, Issues 2008-Q1 and 2008-Q3

17.1.1. „Noise“ -- where does it come from?

That can be answered very quickly and precisely: in each electrical resistance where a current flows and electrons move, there exists noise. As soon as heat comes into play (that is always the case above absolute zero), electrons have an independent existence. They move ever more randomly and not directly from minus to plus. They collide, rebound, are hurled forward or off to the side….. This makes the current vary irregularely by small amounts due to the influence of heat. This effect is called thermal noise. Even if no outside voltage is applied these independent movements of the charge carriers, due to heat, develop a small open circuit voltage V_{noise}. This can be computed as follows:

$$V_{noise} = \sqrt{\frac{4hfBR}{e^{\frac{hf}{kT}} - 1}}$$

Where
h = Planck's constant
k = Boltzmann's constant = 1,38 x 10^{-23} J / Kelvin
T = absolute temperature in Kelvin
B = bandwidth in Hz
f = center frequency of the band in Hz
R = resistance value in Ω

This seems terribly complicated and not practical, but the following simplification can be used without problem to at last 100GHz and temperatures down to 100K:

$$V_{NOISE} = \sqrt{4kTBR}$$

Changing this formula around it suddenly looks much simpler:

$$\frac{\left(\frac{V_{NOISE}}{2}\right)^2}{R} = kTB$$

a) This is a simple indication of power! Each resistance, independent of its resistance value, produces an „Available Noise Power“ proportional to kTB.

b) The resistance should be regarded as voltage source consisting of V_{NOISE} and a noise free *internal resistance* R to generate the noise power. A noise free load resistance with the same value R is connected to this source. Because voltage across the load resistance is then half V_{NOISE} the load resistance receives the Available Noise Power kTB.

c) This noise power increases linearly with the absolute temperature of the circuit and the voltage with the square root of the power. The spectral power density (power per Hz) is independent of the frequency. This is called "White Noise".

Important:
Receiver systems are nearly always specified in terms of „power levels“ instead of voltages. These are logarithmic measurements so that the gain can be calculated by adding levels instead of multiplication. The most familiar unit is “dBm” -- that is not a voltage but a power rating in relation to the system reference resistance:

P_0 = 1 milliwatt at the system resistance

Thus

Power level in dBm = 10log (power value / 1mW)

If the noise power “kTB” is considered in more detail, an interesting simplification can be introduced:

kTB = (kT) x B = (Noise Power Densitiy) x (Bandwidth)

The Noise Power Density „kT“ represents the power in every Hz; this must be multiplied by the bandwidth in order to calculate the noise power produced. Converting this to a power level calculation the following formula should be committed to memory:

Each resistance produces at ambient temperature (T_0 = 290K) an available noise level and thus an available noise power density of

-174dBm per Hz of bandwidth

For a bandwidth larger than 1 Hz:

Available noise power level in dBm = -174dBm + 10log(bandwidth in Hz)

As a simple example:
The no load noise voltage V_{NOISE} at the terminals of a 50Ω resistor for a bandwidth of 100kHz will be calculated:

Noise level with matching = -174dBm + 10log(100000) = -174dBm + 50dB = -124dBm

That results in an available noise power of

$$\mathbf{P = 1mW \bullet 10^{\frac{-124}{10}} = 1mW \bullet 10^{-12,4} = 4 \bullet 10^{-16}\,W}$$

That will produce a voltage across the 50Ω resistor of:

$$\mathbf{V_{NOISE} = \sqrt{P \bullet R} = \sqrt{50\Omega \bullet 4 \bullet 10^{-16}\,W} = 141nV}$$

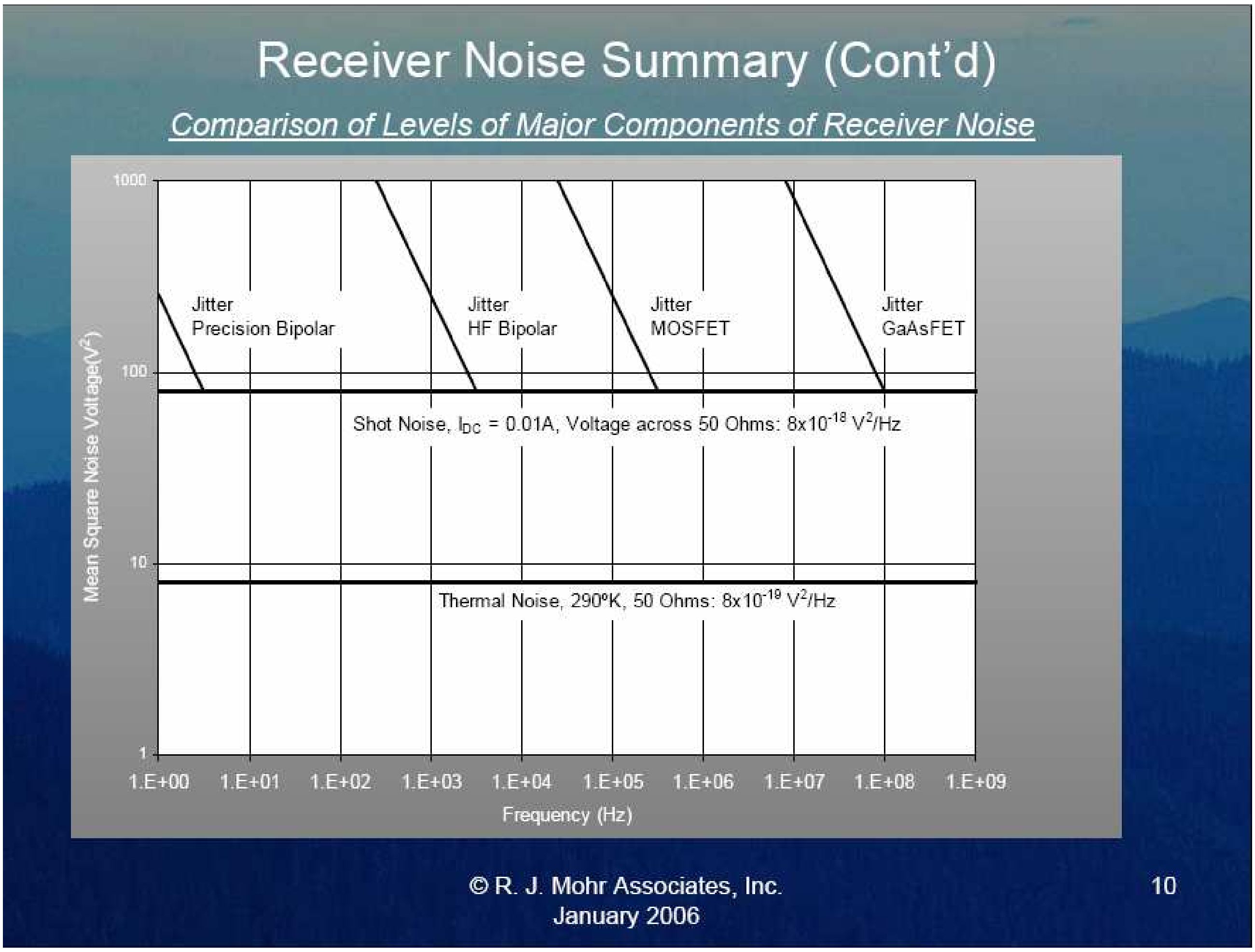
So the open circuit voltage is twice this value, thus 282nV.

17.1.2. Other Sources of Noise

Each active component (valve, bipolar transistor, barrier layer FET, MOSFET, HEMT, etc.) produces two additional types of noise:

a) ***Shot-Noise*** is produced in valve diodes and PN transitions by distortions of the current flow when crossing potential differences . It produces wide band, white noise.

b) ***Flicker-Noise*** or „1 / f – Noise“ (= sparkling noise) results from defects in the crystal structure caused by impurities. They lead to short pulse type fluctuations in the current flow that produce a spectrum whose power density decreases with rising frequency. There exists a corner frequency and it is interesting to see how this differs between active components. The following figure is a good representation. The reference material in the homepage of Mohr Associates should be obtained from the Internet. It contains a precise but compact introduction to the subject.



Receiver Noise Summary (Cont'd)

Comparison of Levels of Major Components of Receiver Noise

© R. J. Mohr Associates, Inc.
January 2006

10

Before semiconductors, gas discharge tubes were used as sources of noise, they produced a very wide band plasma noise.

The following section shows how the different kinds of noise in a circuit can be considered and how they can be summarized by only one parameter.

17.1.3. Noise Temperature and Noise Factor for a Twoport System

When components of a communicatons system are specified they usually have the same system resistance (normally 50Ω, but for radio, television and video systems it is 75Ω) and signal quality information required for correct transmission. This is often expressed by the SINAD value (= Signal to Noise and Distortion). The maximum signal level is limited by over modulation, distortion, intermodulation etc. The "noise floor" (in addition, other disturbances e.g. cross modulation from other channels) gives the minimum signal level in the system. Therefore each component must be specified by its S-Parameters as well as its noise behaviour.

When referring to noise, the following terms are usually evaluated (unfortunately not all authors use the same terminology):

a) The **„equivalent noise temperature T_e"** of a component expresses the self-noise of the component by an additional temperature rise for the noise source resistance. The component itself is thought of as noise free.

b) The **„NOISE FIGURE (NF)"** indicates, in dB, how much worse the signal-to-noise ratio becomes after the signal has passed through the component.

c) The **„NOISE FACTOR (F)"** is like the Noise Figure as above, BUT NOT IN dB, only as a simple power relationship.

17.2. Simulation of the Spectral Noise Density

This sounds complicated, but isn't. In the introduction we stated that every resistor delivers at his terminals an Available Noise Power of:

kTB = (kT) x B = (Noise Power Densitiy) x (Bandwidth)

The noise power densitiy is the power which can be measured in every Hz of the bandwidth. You get the total available power if you multiply this density by the valid bandwidth (…or by integration over the bandwidth if the noise power density varies with the bandwidth).

But very often one prefers to handle the voltage instead of the power -- because it is easier to measure voltages at different frequencies than power. So use the well known equation to jump between these two values:

$$\mathbf{Power} = \frac{\mathbf{Voltage}^2}{\mathbf{Resistance}}$$

by using the known system resistance „R".

When You now calculate the square root of the Noise Power Density, you'll get an expression for the „Spectral Noise Voltage Density", usually given in

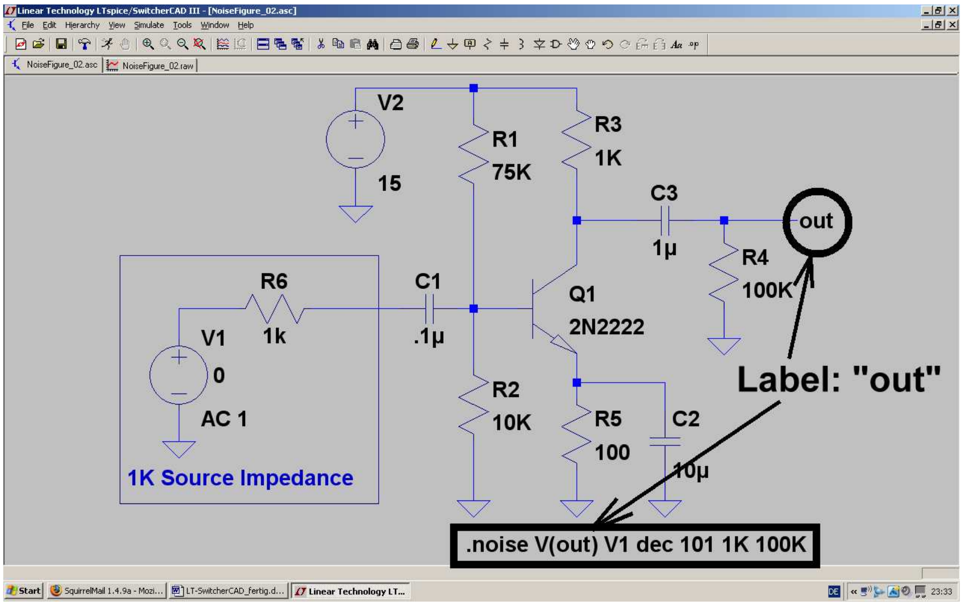
$$\frac{\mathbf{Nanovolts}}{\sqrt{\mathbf{Hz}}}$$

If you multiply this result with the factor

$$\sqrt{\mathbf{Bandwidth}}$$

you'll get the total noise voltage.

Let use consider the included example „**NoiseFigure.asc**" of the software -- you find it in the „Edudational" folder and it is a one stage transistor audio amplifier, which we will modify a little bit for our purposes.


Linear Technology LTspice/SwitcherCAD III - [NoiseFigure_02.asc]
File Edit Hierarchy View Simulate Tools Window Help
NoiseFigure_02.asc
NoiseFigure_02.raw
V2
15
R1
75K
R3
1K
C3
1µ
out
R4
100K
R6
1k
C1
.1µ
Q1
2N2222
V1
0
AC 1
R2
10K
R5
100
C2
10µ
1K Source Impedance
Label: "out"
Start
SquirrelMail 1.4.9a - Mozi...
LT-SwitcherCAD_fertig.d...
Linear Technology LT...
23:33

```
.noise V(out) V1 dec 101 1K 100K
```

At the output we add the **label „out**“ for the noise simulation.

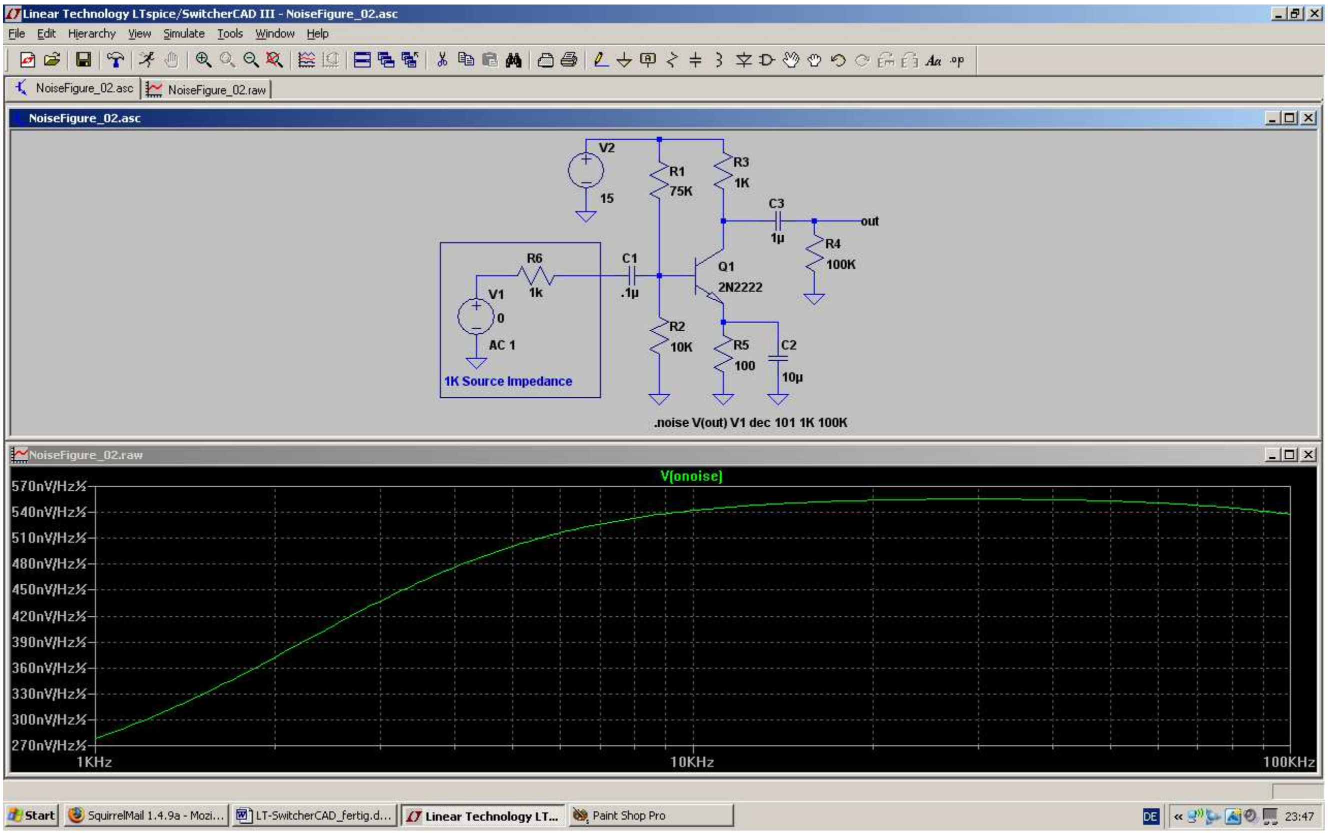
Use the path „**Simulation / Edit Simulation Command / Noise**“ to edit the necessary simulation command:

.noise V(out) V1 dec 101 1k 100k

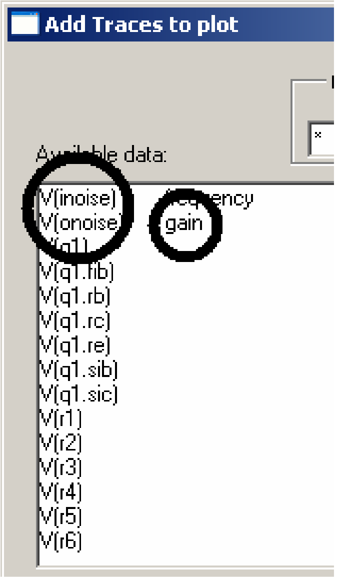
This means:

Simulate the spectral noise voltage density at the label „out“ and refer it to voltage source V1 with internal resistor R6 = 1k.
Use a decadic sweep with 101 points per decade from 1kHz to 100kHz.

And this is the result for the output noise voltage density V(onoise) when clicking with the „cursor probe“ on the output pin:



Please examine the scale and the units of the vertical axis....

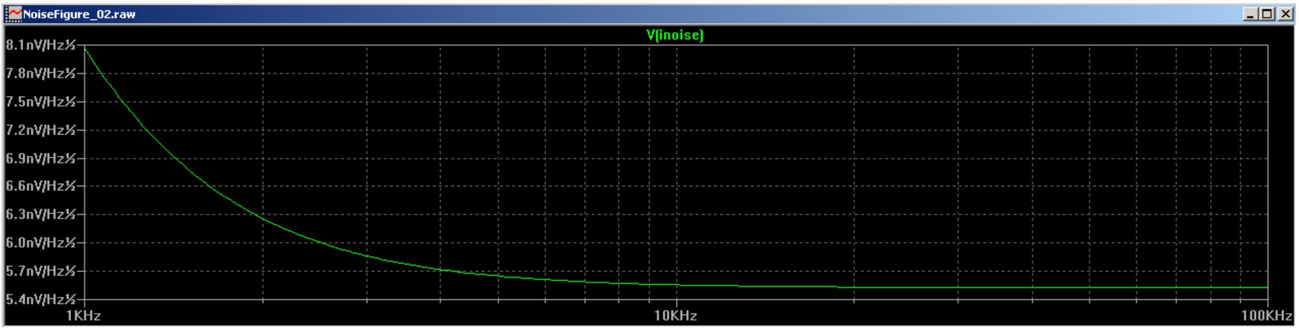


No problem, now to show the **noise voltage density V(inoise), referred to the input!**

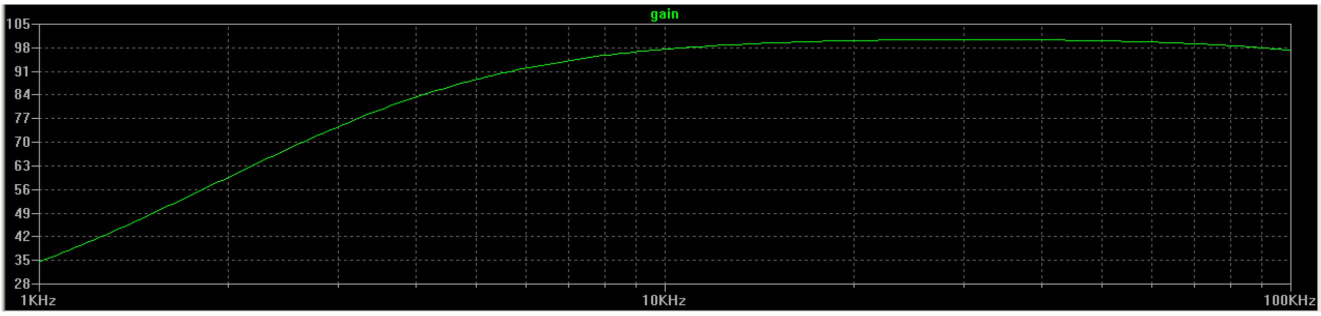
This is a voltage source which can be thought in series to the noise free source V1. So right click on the diagram, choose „**Add Trace**“ and then „**V(inoise)“** in the list (see left figure).

If you examine the list then you‘ll find that not only V(inoise) can be shown but also V(onoise) and the **gain.**

This is the **input noise voltage density V(inoise) …..**



…and this is the **gain versus frequency**:



17.3. Simulation of the Noise Figure NF (in dB)

In communications systems many blocks are connected in series. So the reduction in signal quality in every „twoport“ caused by noise must be known and is expressed by the

Noise Figure NF in dB

So the „**NOISE FIGURE (NF)**“ ist the difference in db between the signal to noise ratio at the input and the output of the twoport.
For a noise free twoport the signal to noise ratio is identic at input and output pin and NF = 0dB

(Please note:
The input signal source V1 delivers not only the desired communication signal to the twoport but also noise due to the internal resistance of the source. So if only this noise is amplified by the same factor as the communicaton signal, the signal to noise ratio at the output would stay the same as at the input. But because the block adds noise itself....)

If you change the NF-value in dB to a linear ratio, then you get the „noise factor F“.

Now let us tackle the NF-simulation with LTSpice, following these steps:

a) To get „dB“ at the result, we need the relationship

$$\mathbf{NF = 10 \times \log_{10}(.....)}$$

b) In the brackets we must insert the ratio of the total input noise power (= total output noise power, refered back to the input = P_{inoise}) to the noise power part which comes only from the source resistance (= P_{Source}). If both powers are equal, then we have a noise free twoport and the noise figure NF is

$$\mathbf{NF = 10 \times \log(1) = 0\ dB}$$

But in reality you will always have noise factor values larger than 1 = NF-values larger than 0 db....

The noise power produced by a resistor increases linearly with the absolute temperature, and T = 290K was always the standard value for every noise calculation. But now the global temperature is rising, climate is changing and so electronic people prefer calculations with T = 300K.
Now let us continue with the calculation of the power ratio:

$$\frac{P_{inoise}}{P_{Source}} = \frac{\left(\frac{V_{inoise}^{2}}{4 \bullet R_{Source}}\right)}{k \bullet T \bullet B} = \frac{V_{inoise}^{2}}{4 \bullet k \bullet T \bullet B \bullet R_{Source}}$$

„k“ is the Boltzmann constant, „T“ is the absolute temperature in Kelvin, „B“ ist he bandwidth of the system, R_{Source} is the internal resistance of the input voltage source.

If we consider spectral power density then the bandwidth is automatically 1Hz. Then we can ignore the bandwidth “B” for the calculation and get the final formula which must be converted to a SPICE command:

$$NF = 10 \bullet \log\left(\frac{V_{inoise}^{2}}{4 \bullet k \bullet T \bullet R_{Source}}\right) \text{ in dB}$$

And assuming a actual climate change contribution we use an absolute temperature of 300K:

$$\mathbf{NF} = \mathbf{10 \bullet \log\left(\frac{V_{inoise}{}^{2}}{4 \bullet k \bullet 300 \bullet R_{Source}}\right)} \text{ in dB}$$

Problem:

In a simulation with LTSpice it is only possible to calculate this formula (and to show the result) by the usage of a

PLOT command!

So we must write the following line for the calculation of the noise figure NF(1K) with a text editor, when the source resistor’s value is 1k:

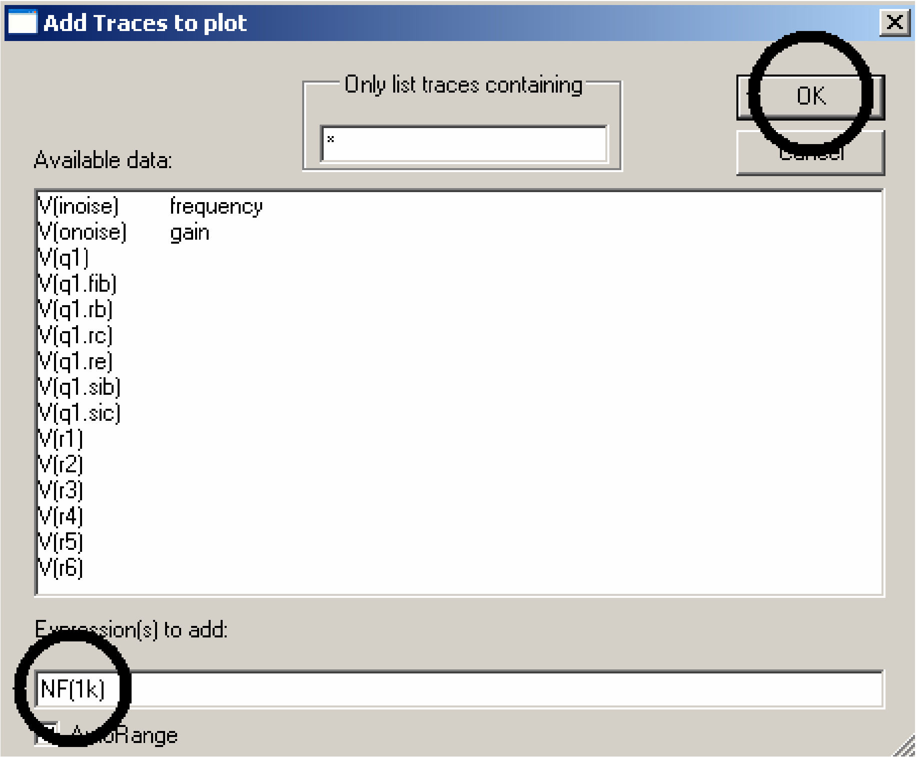
```
.func   NF(1k)    10 * log10(V(inoise) * V(inoise) / (4*k*300*1k))
```

Save this line in a new file named

plot.defs

and save it in the LTSpice program folder.

Caution: After saving this file you must close LTSpice and start it again. Only in this manner the new „plot.defs“ are included and used by the software!



Only list traces containing: *

Available data:

```
V(inoise)    frequency
V(onoise)    gain
V(q1)
V(q1.fib)
V(q1.rb)
V(q1.rc)
V(q1.re)
V(q1.sib)
V(q1.sic)
V(r1)
V(r2)
V(r3)
V(r4)
V(r5)
V(r6)
```

Expression(s) to add: NF(1k)

The rest is simple: simulate the circuit, click right on the result diagram and choose

„Add Plot Pane“

to get an empty diagram.

Then click right on this new diagram an call

„Add Trace“

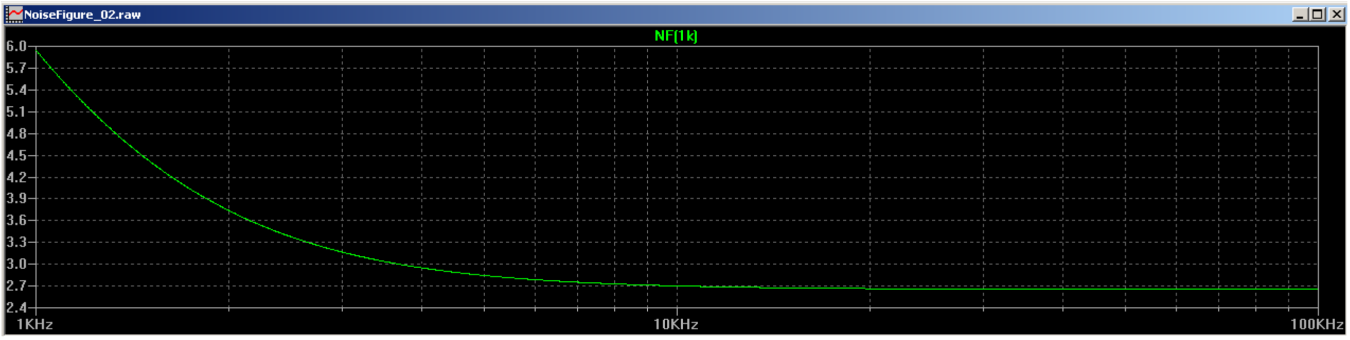
Enter

NF(1k)

In the empty command line and press OK

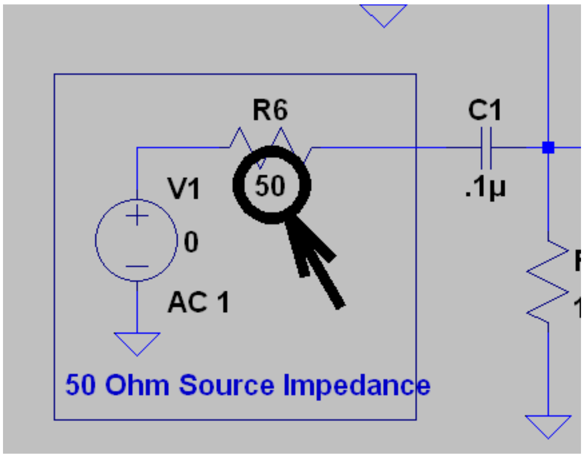
The result is not totally perfect, because

the „dB“ calibration on the vertical axis is missing and this cannot be altered ...



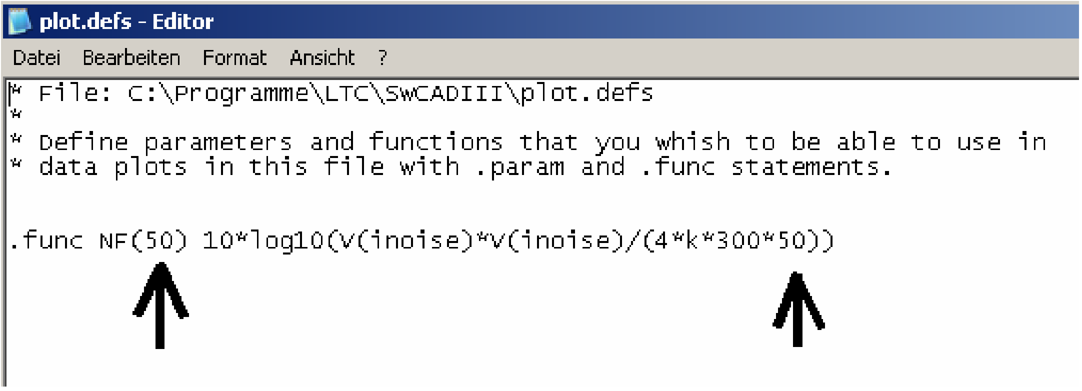
Task:
Communications systems generally use a system resistance of 50Ω. So please repeat the NF simulation for this value.

Step 1:



Change the internal resistance of the voltage source V1 in the schematic to 50Ω.

Step 2:



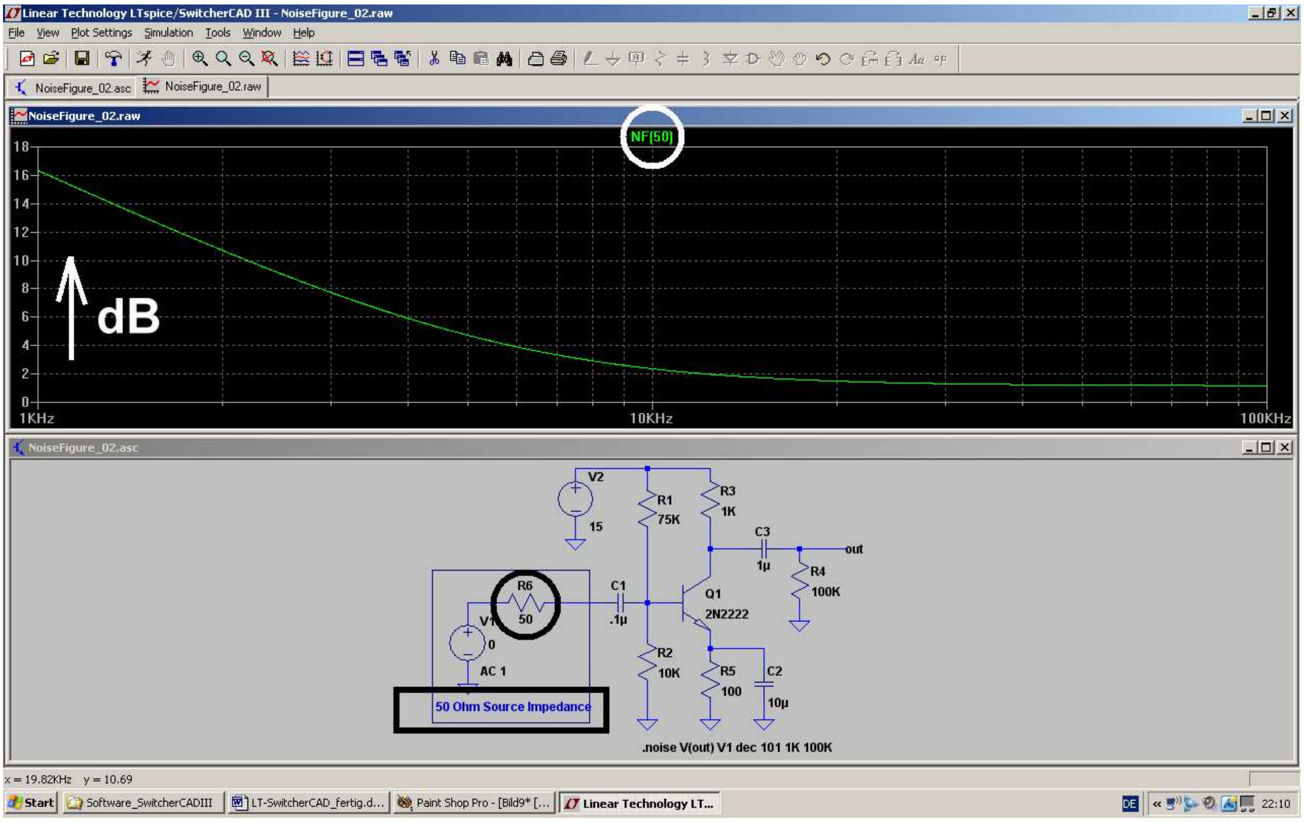
```
* File: C:\Programme\LTC\SwCADIII\plot.defs
*
* Define parameters and functions that you whish to be able to use in
* data plots in this file with .param and .func statements.

.func NF(50) 10*log10(v(inoise)*v(inoise)/(4*k*300*50))
```

Open the file „**plot.defs**“ with the text editor and modify the command line to „**NF(50)“** as shown in the figure. Do not forget to save this modified file correctly…

Caution:
Please close LTspice now and start it again. Only then the modification is accepted and used by the software. Otherwise you get a cryptic error message…

Step 3:
Simulate again, open a new „Plot Pane“ and then “Add Trace**”. Enter “NF(50)” in the line „Expression(s) to add“ and press OK. This is the result:**

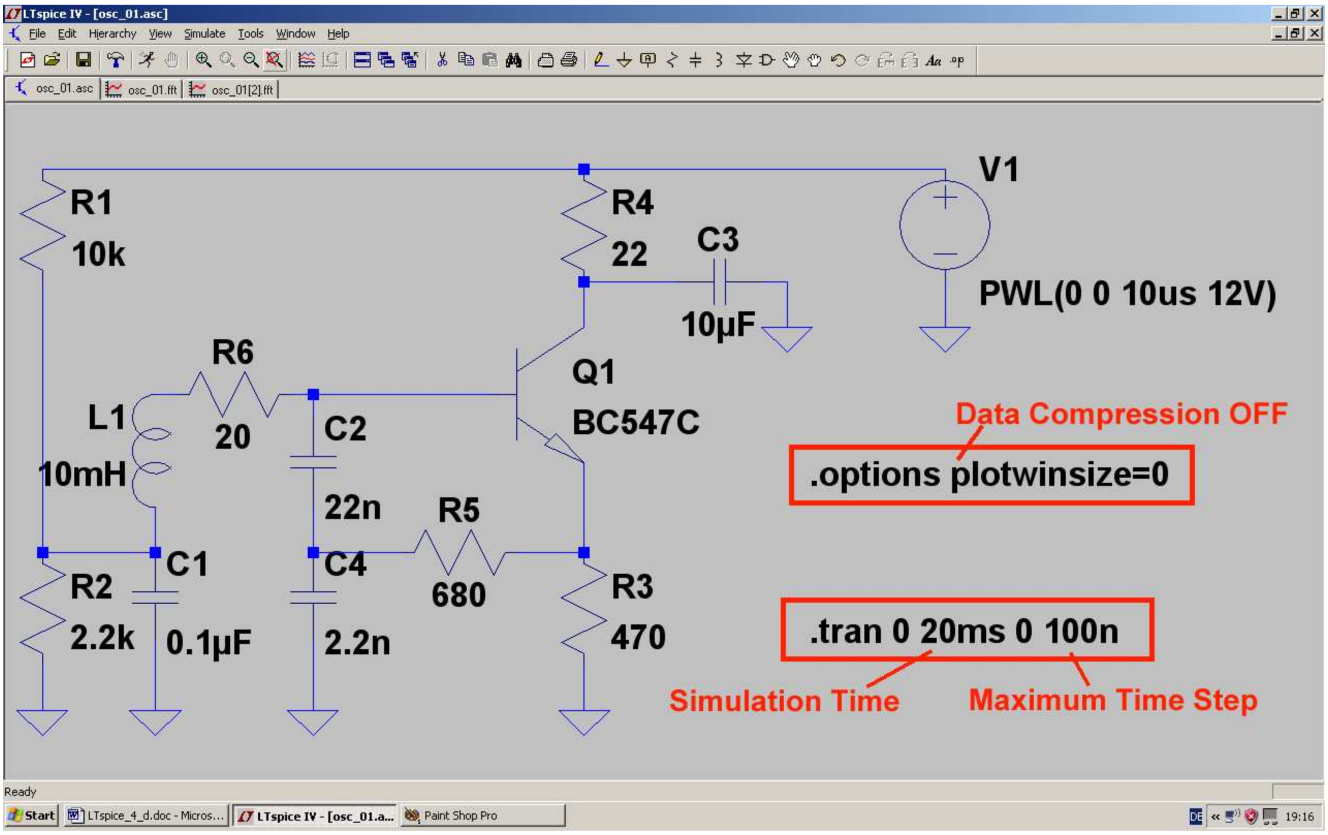


.noise V(out) V1 dec 101 1K 100K

The vertical axis is now calibrated in „dB“, but this is not indicated….

18. Simulation of a Sine Oscillator

Let us have a look at a Colpitts sine oscillator using a collector-grounded NPN – transistor.
This is the schematic:



Data Compression OFF

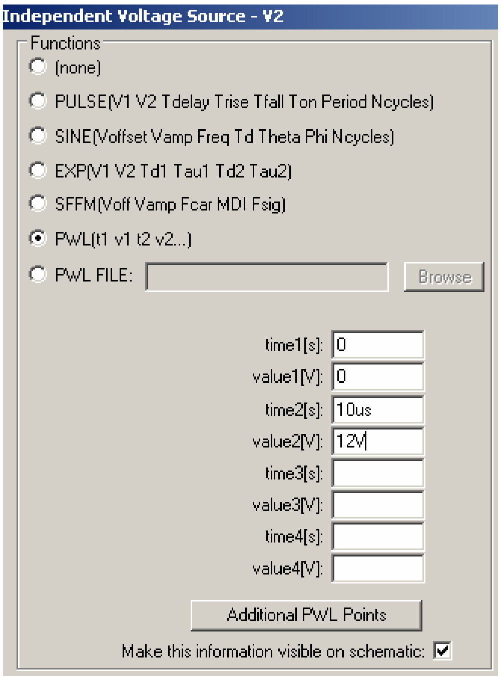
.options plotwinsize=0

.tran 0 20ms 0 100n

Simulation Time

Maximum Time Step

Note:


Independent Voltage Source - V2
Functions
○ (none)
○ PULSE(V1 V2 Tdelay Trise Tfall Ton Period Ncycles)
○ SINE(Voffset Vamp Freq Td Theta Phi Ncycles)
○ EXP(V1 V2 Td1 Tau1 Td2 Tau2)
○ SFFM(Voff Vamp Fcar MDI Fsig)
○ PWL FILE:
Browse
Additional PWL Points

● PWL(t1 v1 t2 v2...)

time1[s]: 0
value1[V]: 0
time2[s]: 10us
value2[V]: 12V
time3[s]:
value3[V]:
time4[s]:
value4[V]:

Make this information visible on schematic: ☑

To start the (simulated) oscillation the circuit needs a little „push". This is done by programming a special property of the supply voltage:

„**Rise linearly from 0V to +12V in a rise time of 10 microseconds".**

We right click on the symbol of the voltage source, choose „**Advanced**" and use the PWL-function with the value pairs:

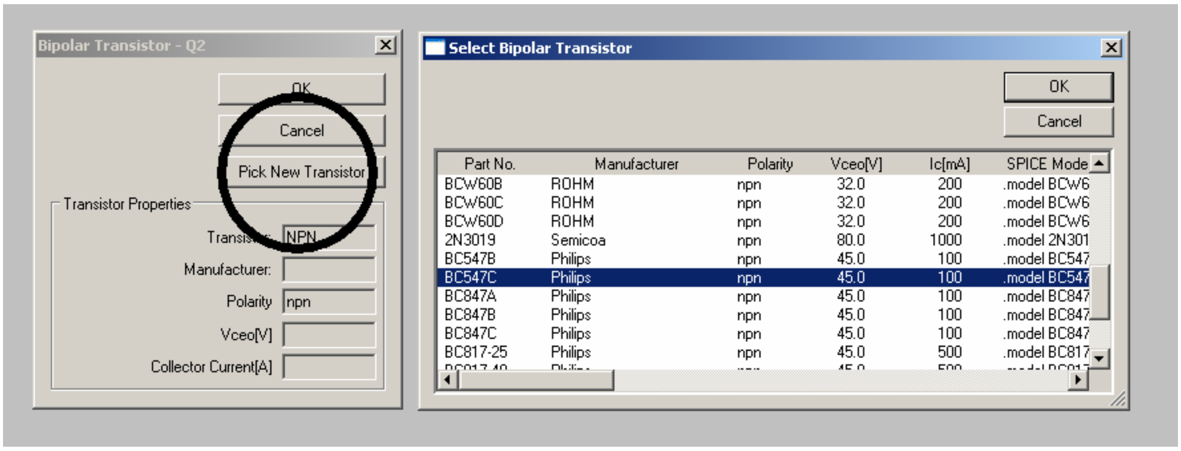
0V at t = 0

+12V at t = 1us

The „Simulation Settings" are set to „Time domain (Transient)" and the simulation time is **0….20ms. This gives a frequency resolution of 1/20ms = 50Hz in the spectrum.** The maximum time step is set to 100ns, so SPICE doesn't overlook the starting procedure described above. . This now gives a number of 20ms / 100ns = 200 000 sample for the FFT computation.

And the data compression must be switched OFF by the directive **.options plotwinsize=0**

Now we need a transistor with the symbol name „**npn**“ from the part library. Search and place it on the schematic. As soon as you right click on the NPN transistor’s symbol, press the „**Pick New Transistor**“ button. Then find the **BC547B** which we want to use for our simulation**.**



Transistor Properties

Transistor: NPN

Manufacturer:

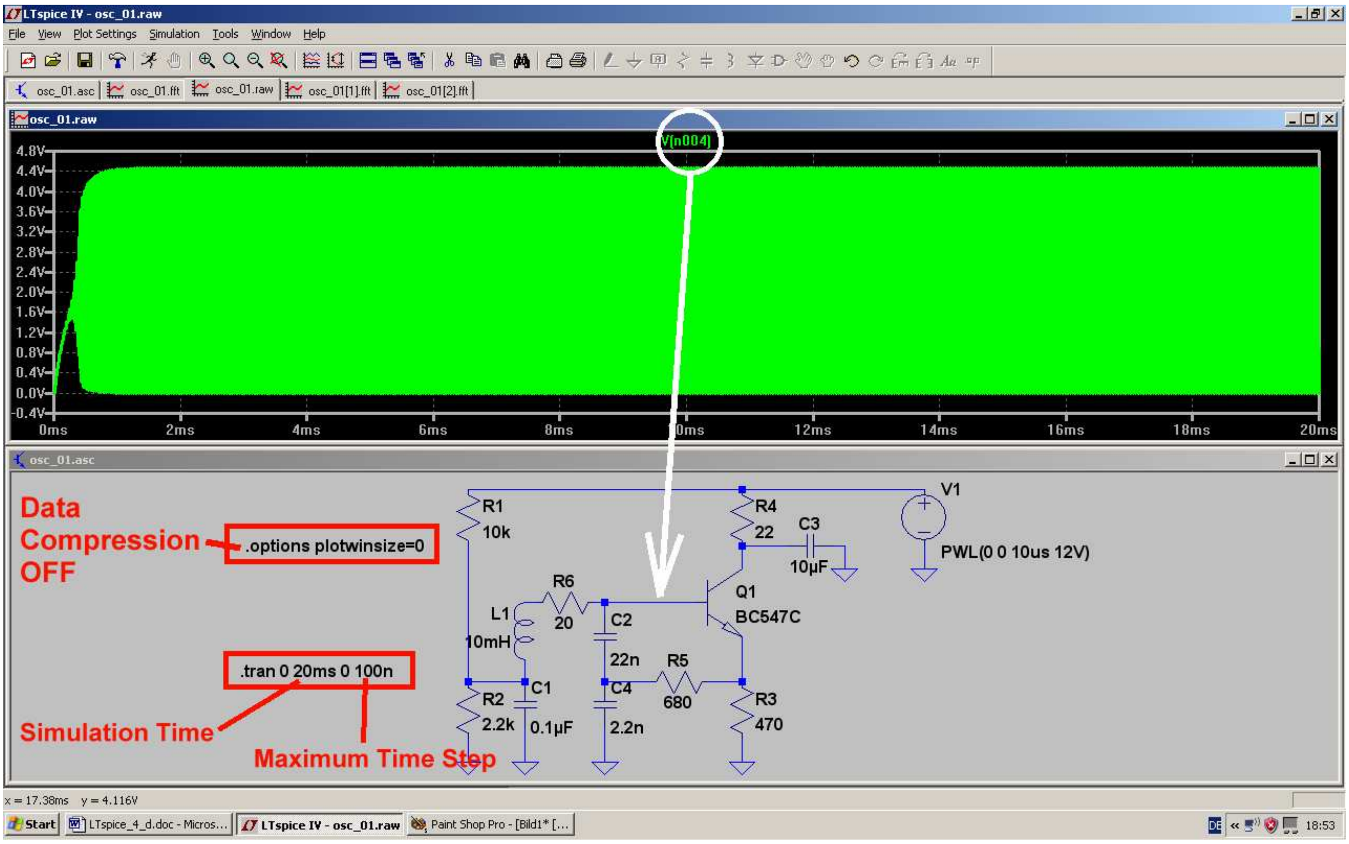
Polarity: npn

Vceo[V]:

Collector Current[A]:

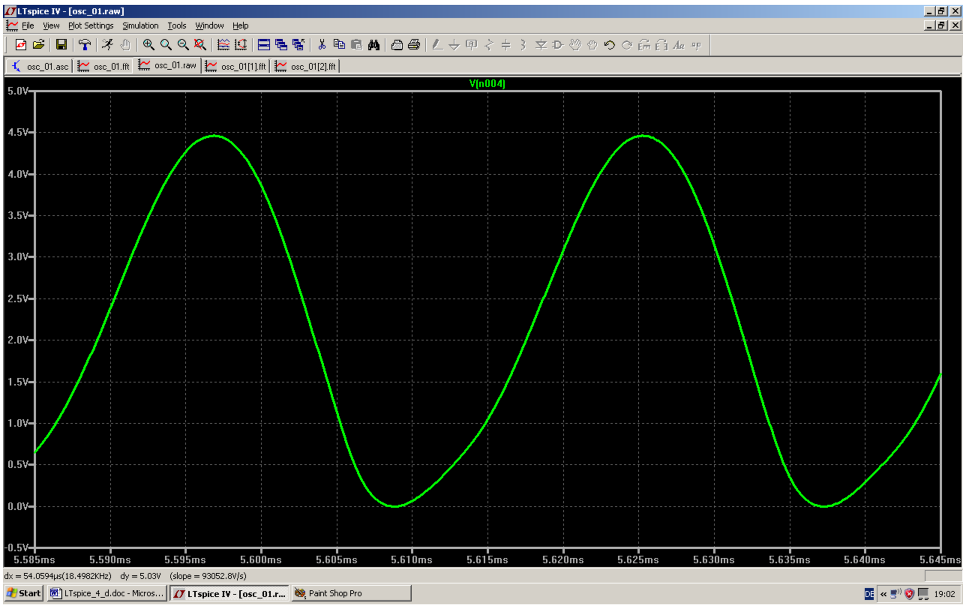
Part No.	Manufacturer	Polarity	Vceo[V]	Ic[mA]	SPICE Mode
BCW60B	ROHM	npn	32.0	200	.model BCW6
BCW60C	ROHM	npn	32.0	200	.model BCW6
BCW60D	ROHM	npn	32.0	200	.model BCW6
2N3019	Semicoa	npn	80.0	1000	.model 2N301
BC547B	Philips	npn	45.0	100	.model BC547
BC547C	Philips	npn	45.0	100	.model BC547
BC847A	Philips	npn	45.0	100	.model BC847
BC847B	Philips	npn	45.0	100	.model BC847
BC847C	Philips	npn	45.0	100	.model BC847
BC817-25	Philips	npn	45.0	500	.model BC817

Simulate the voltage at the base of the transistor to get this screen:



```
.options plotwinsize=0
.tran 0 20ms 0 100n
```

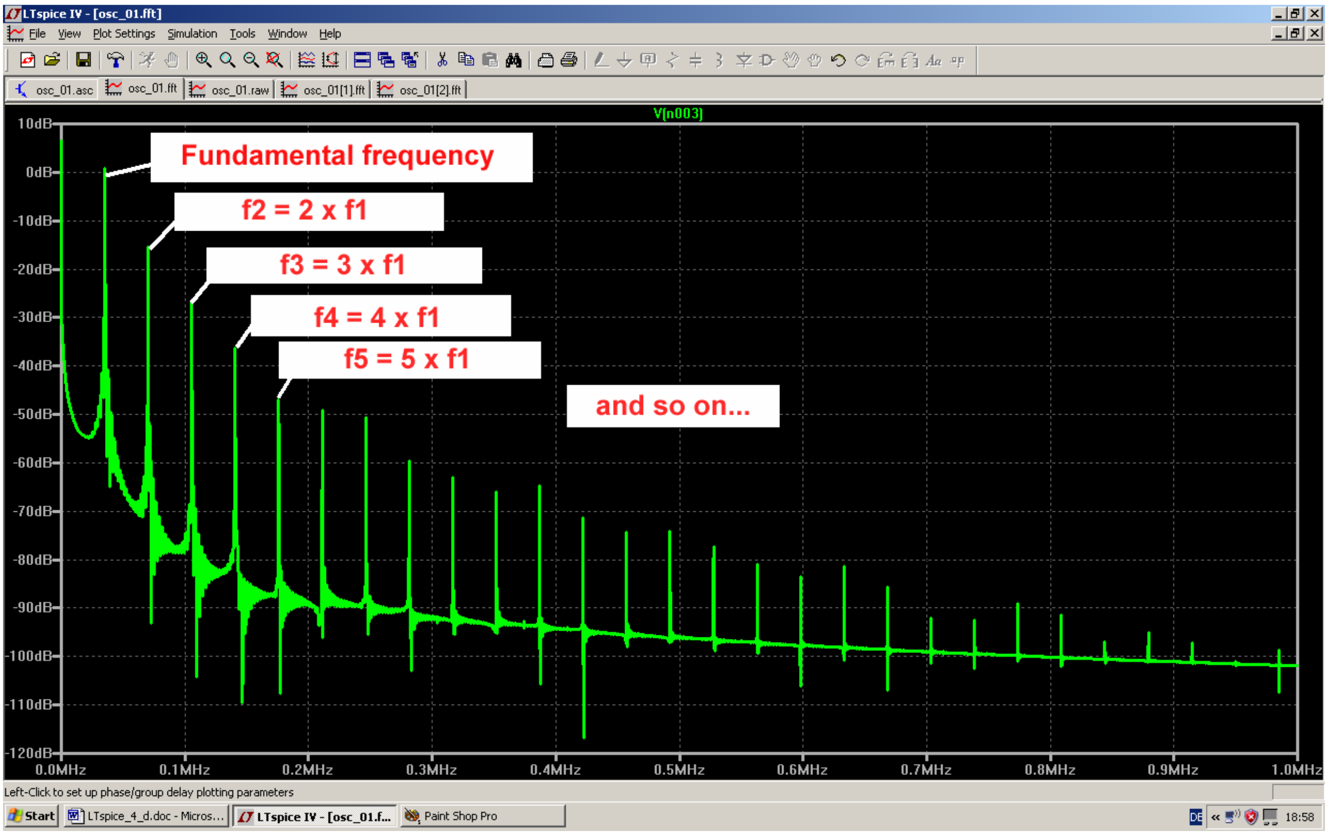
Use the zooming function to show the details of this voltage and to determine the time period for one cycle of the signal. Calculate the oscillating frequency. Examine the curve for distortion.



dx = 54.0594µs(18.4982KHz) dy = 5.03V (slope = 93052.8V/s)

The duration of the period is 28ms and this gives a frequency of 35,7kHz.
But the curve is a little “clipped” on one side and this means that the feedback is too “strong”. This can be seen when viewing the **frequency spectrum** of this signal.

So right click on the waveform viewer window and choose „View“, then „FFT“ with 131 072 data points in time.
At lastly show the curve with a linear frequency range from **0 to 300 kHz with a tick of 50 kHz**:



Left-Click to set up phase/group delay plotting parameters

Now we can see the decreasing amplitudes of the harmonics with increasing frequency. Also the “fundamental oscillating frequency“ can now be determined more accurately as 35.7kHz.

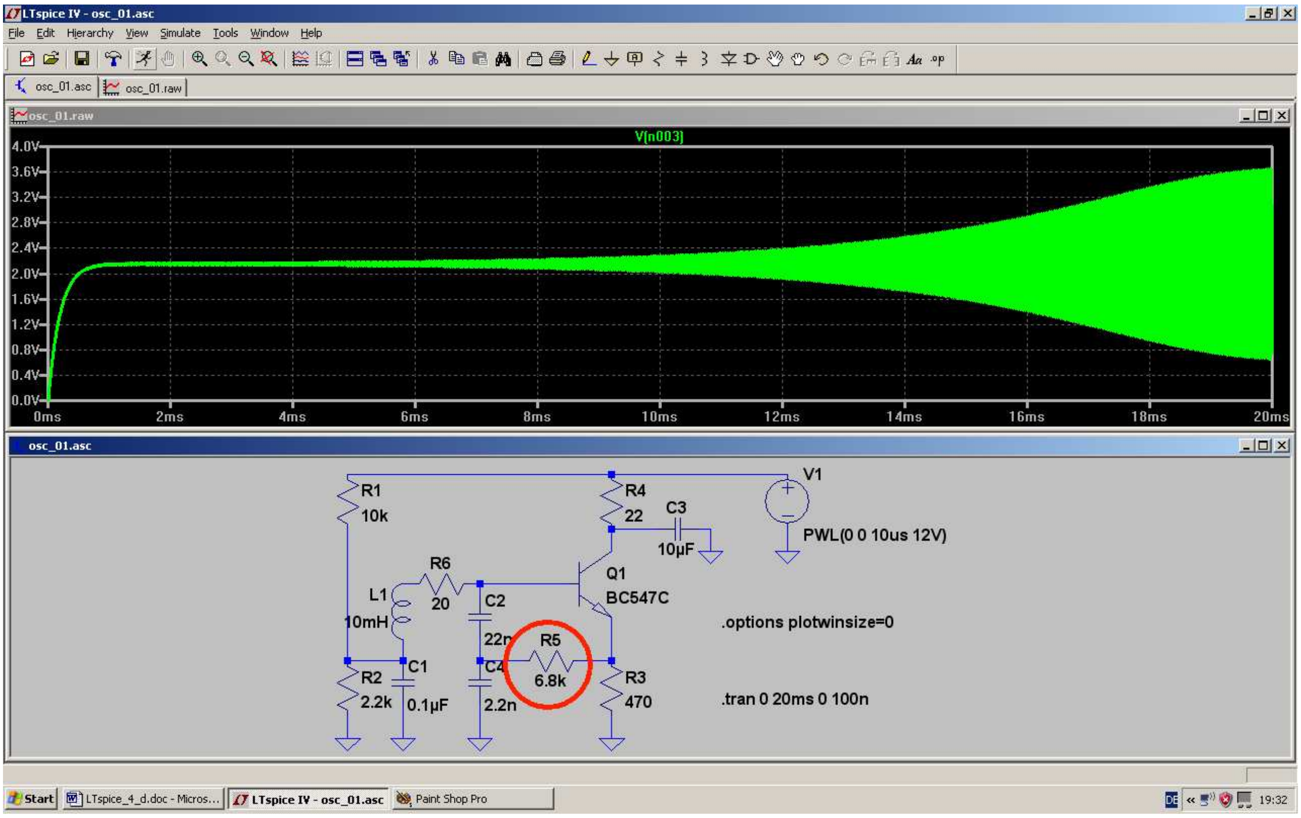
But the 2nd harmonis is only attenuated by less than 20dB and this is caused by the unsymmetry of the signal (Remember: in a really symmetric signal you don’t find any even harmonics….).
So any distortion of the symmetry can at once be recognized by testing for even harmonics in the spectrum.

Note:
Resistor R4 / 680Ω gives the positive feedback which causes the oscillation. A part of the emitter AC current is flowing into the resonant circuit through this resistor and „compensates the losses“.
So if you increase the resistor’s value, the positive feedback will get weaker and weaker. This gives less distortion but the oscillation will suddenly stop.

Task:
Try to find out this stop point by increasing the value of R4 and repeating the simulation.
Watch the improvement of the waveform and check the better attenuation of the harmonics by the FFT-simulation of the frequency spectrum..

Also be aware of the „oscillation start delay“ when reducing the feedback. When approaching the critical point you get signficantly longer delays before the oscillation starts.

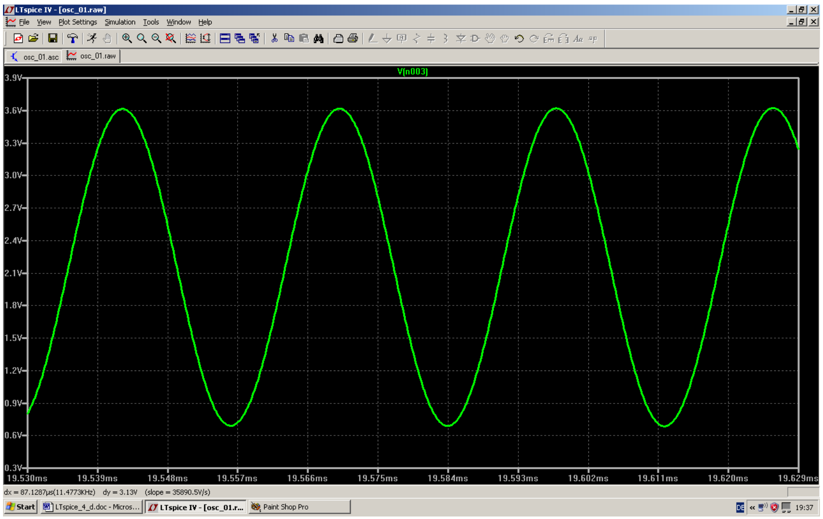
(Note: with a resistor value of 6.8k you are not far from this point…please check it!)



.options plotwinsize=0

.tran 0 20ms 0 100n

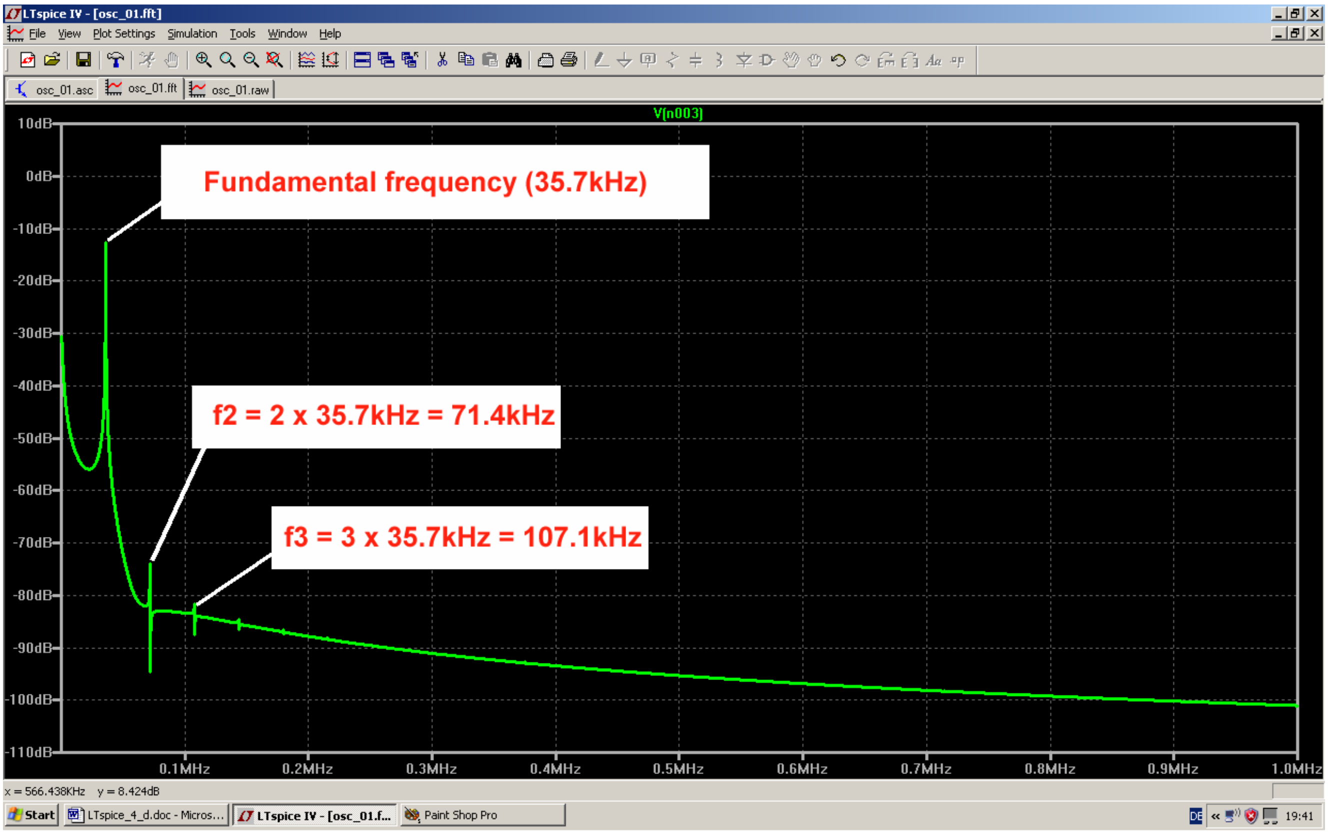
This is the result for 6.8k. The oscillation start delay is increasing….



dx = 87.1287µs(11.4773KHz) dy = 3.13V (slope = 35890.5V/s)

…but now you get
a nearly perfect curve!

And the FFT spectrum proves this.



LTspice IV - [osc_01.fft]
File View Plot Settings Simulation Tools Window Help
osc_01.asc
osc_01.fft
osc_01.raw
V(n003)
10dB
0dB
-10dB
-20dB
-30dB
-40dB
-50dB
-60dB
-70dB
-80dB
-90dB
-100dB
-110dB
0.1MHz
0.2MHz
0.3MHz
0.4MHz
0.5MHz
0.6MHz
0.7MHz
0.8MHz
0.9MHz
1.0MHz
Fundamental frequency (35.7kHz)
f2 = 2 x 35.7kHz = 71.4kHz
f3 = 3 x 35.7kHz = 107.1kHz
x = 566.438KHz y = 8.424dB
Start
LTspice_4_d.doc - Micros...
LTspice IV - [osc_01.f...
Paint Shop Pro
19:41

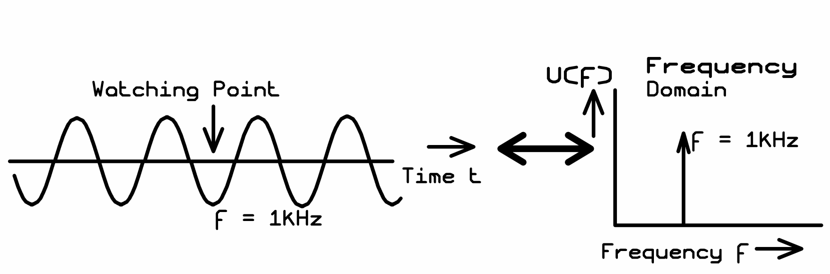
19. Signals and Harmonics

19.1. Fundamentals

If you ask anyone what signals are found on power lines all over the world, you'll always get the same answer: "a sine wave with a RMS-value between 110V and 230V and a frequency of 50 or 60Hz, depending of the specific country". But if you continue and ask:

"Is that all?"

Even specialists may give a wrong answer. OK, there are sometimes harmonics of the line frequency, but much more important are the effects caused by switching on or off. So let us look at the correct definition of a signal consisting of only one frequency:

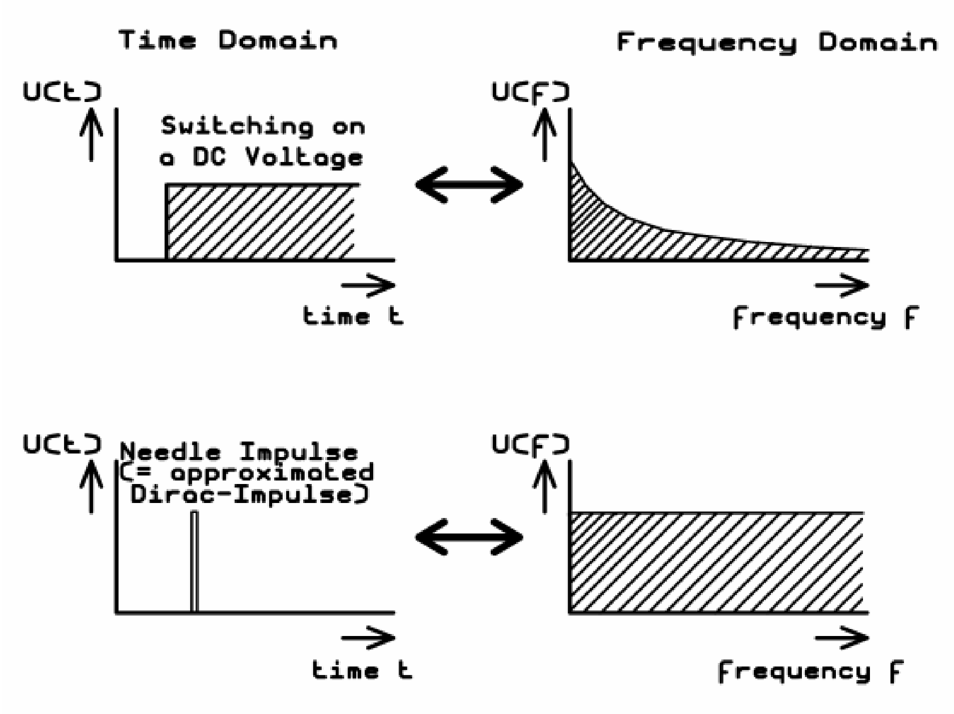


Only a pure continuous sine wave where neither the start point nor the stop point can be observed consists of only one spectral line!

This means that in every day usage you should always clearly distinguish between "**distortions of a signal**" and "**switching events including changes".**

Switching

Every switching ON or switching OFF of a signal or increasing or decreasing the amplitude produces a "signal collection" which travels away (with light velocity) from the point of generation. You know that as a "click" from a transistor radio when switching on the light in a room and you should know that in this moment you get the complete frequency range loaded with energy! Very low and low frequencies will travel away using wires, high frequencies prefer wireless travel. But normally the amplitudes decrease rapidly with frequency when "changing a state very slowly" and so TV and communication are still relatively unaffected.
Two examples:



Switching On or OFF of a DC voltage gives a spectrum with decreasing spectral power density at increasing frequencies.

But:
Very short pulses (with nearly Dirac form) have a constant power density over the complete frequency range.

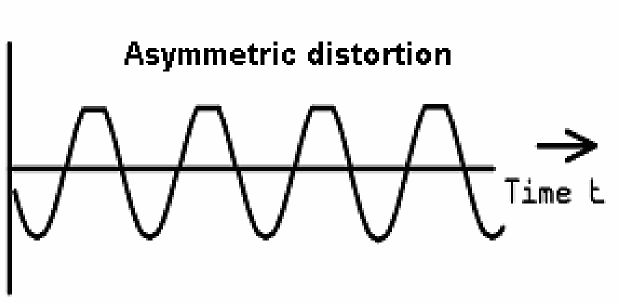
So you cannot avoid this "plop" because there are no gaps in the complete frequency spectrum!

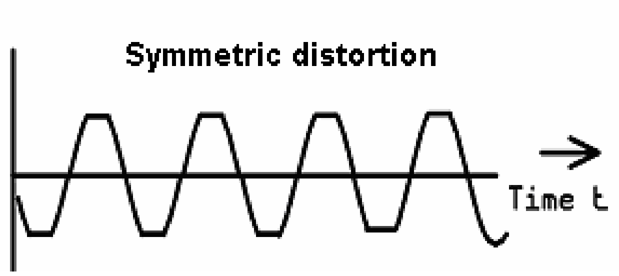
Distortions of periodic signals

Please remember:

Every -- even the smallest! -- deviation from the perfect sine wave generates new additional frequencies!

These new frequencies are called “harmonics” and they are multiples of the fundamental frequency





But there is another axiom:

If only one half of the curve is distorted then we speak of “**asymmetric distortion**” where you **find even and odd harmonics** in the spectrum.
This will also happen when the positive and the negative half is distorted in a different manner.

But when you find perfect symmetrical distortion, the spectrum consists only of the fundamental frequency and ODD HARMONICS!

This means that there are only the lines with

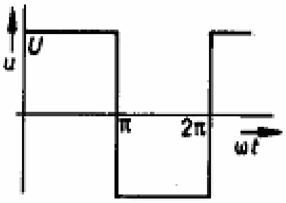
f1 / 3 x f1 / 5 x f1 / 7 x f1...

Note:
**If a sine wave’s amplitude, phase or frequency is varied by another signal, you will also find new frequencies.
But this is the mystery of modulation…..**

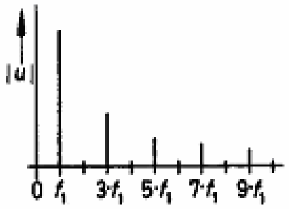
On the next page you find a little overview for different waveforms with the signals in the time domain, in the frequency domain and the formulae to compute the harmonics.

Square Wave

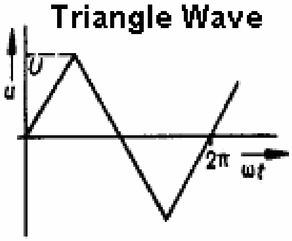
Frequency Domain



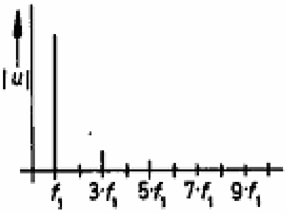
$$f(t) = \frac{4\cdot U}{\pi}\left(\sin \omega\cdot t + \dots\right.$$
$$\left.\dots + \frac{1}{3}\sin 3\,\omega t + \frac{1}{5}\sin 5\,\omega t + \dots\right)$$



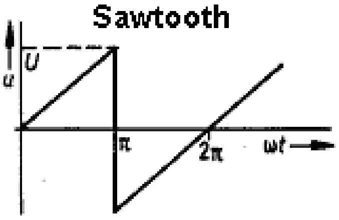
Triangle Wave



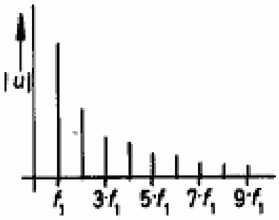
$$f(t) = \frac{8\,U}{\pi^2}\left(\sin \omega\, t - \dots\right.$$
$$\left.\dots + \frac{1}{3^2}\sin 3\,\omega t + \frac{1}{5^2}\sin 5\,\omega t - \dots\right)$$



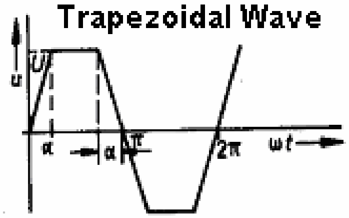
Sawtooth



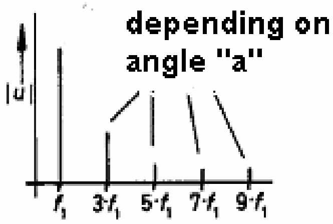
$$f(t) = \frac{2\,U}{\pi}\left(\sin \omega t - \frac{1}{2}\sin 2\,\omega t + \dots\right.$$
$$\left.\dots + \frac{1}{3}\sin 3\,\omega t - \dots +\right)$$



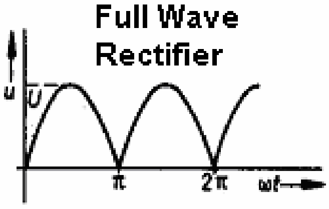
Trapezoidal Wave



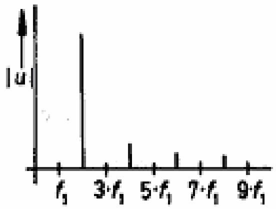
$$f(t) = \frac{4\cdot U}{\pi\cdot\alpha}\left(\sin \alpha\cdot\sin \omega t \underset{\text{tot}}{+} \dots\right.$$
$$\dots + \frac{\sin 3\,\alpha}{3^2}\cdot\sin 3\,\omega t + \dots$$
$$\left.\dots + \frac{\sin 5\,\alpha}{5^2}\sin 5\,\omega t + \dots\right)$$



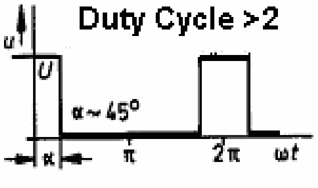
Full Wave Rectifier



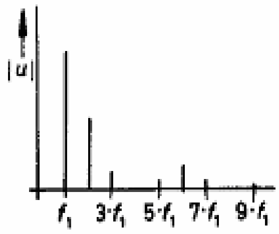
$$f(t) = \frac{4\,U}{\pi}\left(\frac{1}{2} - \frac{1}{1\cdot 3}\cos 2\,\omega t - \dots\right.$$
$$\dots - \frac{1}{3\cdot 5}\cos 4\,\omega t - \dots$$
$$\left.\dots - \frac{1}{5\cdot 7}\cos 6\,\omega t - \dots\right)$$



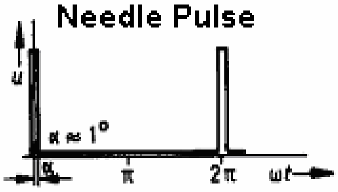
Square Wave Duty Cycle >2

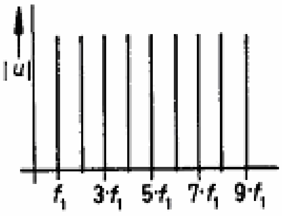


$$f(t) = \frac{2\,U}{\pi}\left(\frac{\alpha}{2} + \frac{\sin \alpha}{1}\cos \omega t + \dots\right.$$
$$\dots + \frac{\sin 2\,\alpha}{2}\cos 2\,\omega t + \dots$$
$$\left.\dots + \frac{\sin 3\,\alpha}{3}\cos 3\,\omega t + \dots\right)$$



Needle Pulse





19.2. Simulation of a Single Pulse Spectrum

Now let us test the statement in the last section and examine the following pulse signal:

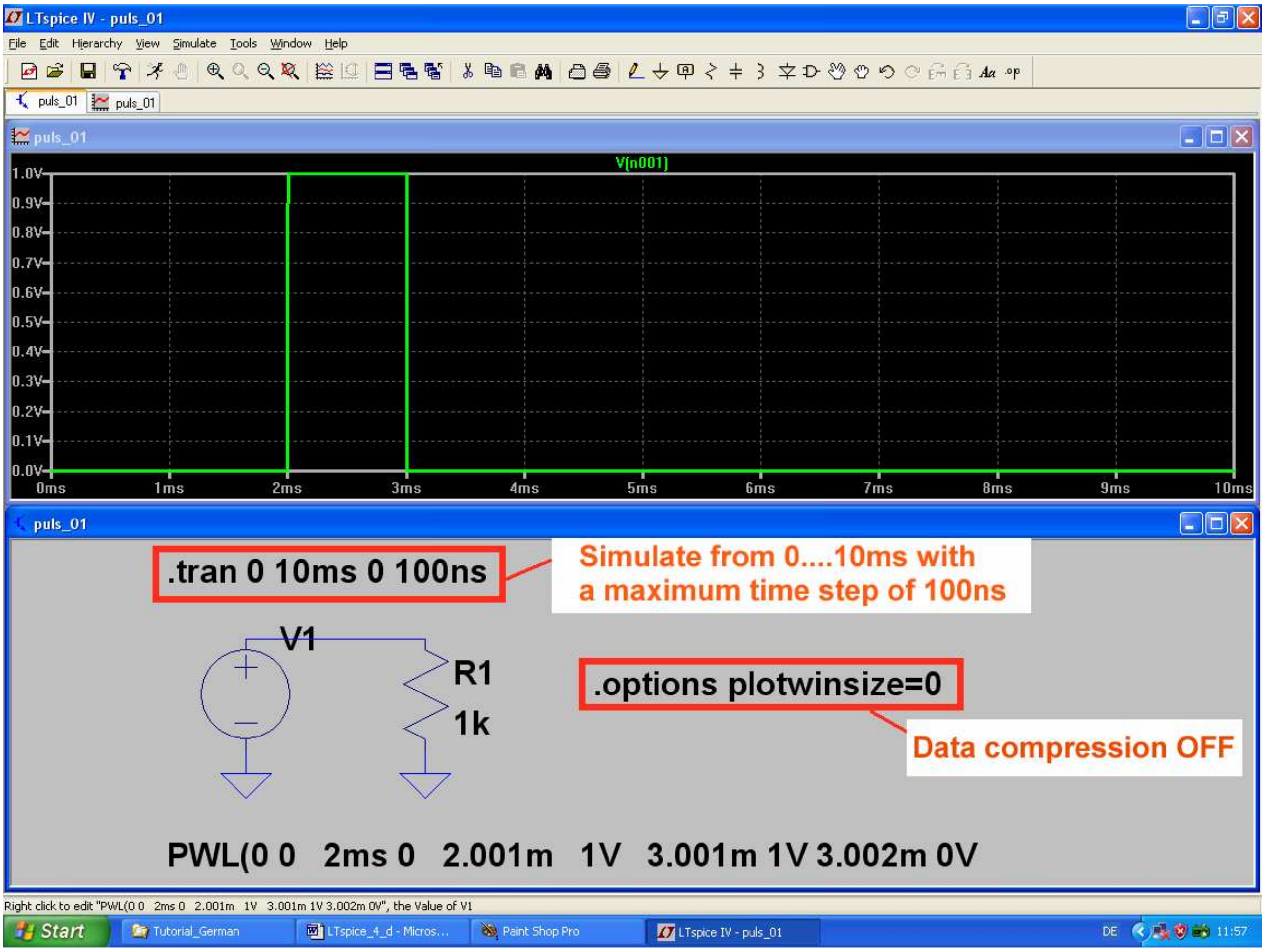
Umin = 0V ; Umax = 1V ; risetime = falltime = 1µs ; pulse width = 1ms.

The pulse signal (measured at R1) starts after a „**delay time“ of 2ms.** The simulation time is **0… 10ms** and the „maximum time step“ has a value of **100ns.** So we get 10ms/100ns = 100 000 real samples of the simulated curve. ***And the data compression of the result file must be switched off by the directive “.options plotwinsize=0”.***

The signal is generated by a **PWL voltage source using the following data:**

0 Volt	**at t = 0**
0 Volt	**at t = 2ms**
1 Volt	**at t = 2.001ms**
1 Volt	**at t = 3,001 ms**
0 Volt	**at t = 3,002 ms**

After the simulation of the voltage at R1 open the menu „Window“ and choose the option **tile horizontally**“. This gives this screen.



.tran 0 10ms 0 100ns

Simulate from 0....10ms with a maximum time step of 100ns

.options plotwinsize=0

Data compression OFF

PWL(0 0 2ms 0 2.001m 1V 3.001m 1V 3.002m 0V

Right click to edit "PWL(0 0 2ms 0 2.001m 1V 3.001m 1V 3.002m 0V", the Value of V1

Now we want to perform a Fast Fourier Transformation and so some explanation is necessary:
Choosing a simulation time of 10ms is crucial for the FFT simulation result because you get from it the lowest frequency of the spectrum AND the frequency resolution (…including the line width of a signal…) by the following relationship:

frequency resolution = 1 / simulation time = 1 / 10ms = 100Hz

Choosing a “maximum time step” of 100ns during this simulation time of 10ms results in a minimum amount of **10ms / 100ns = 100 000 „samples“** which are then saved.

This is the absolute upper limit for the „number of data point samples in time“ which must be entered in the menu when calling and before starting the FFT. So let’s stay on the safe side and use the preset value of 65536 Samples.

If you use more than the mentioned 100 000 samples: this can cause errors and / or an additional noise floor at high frequencies).

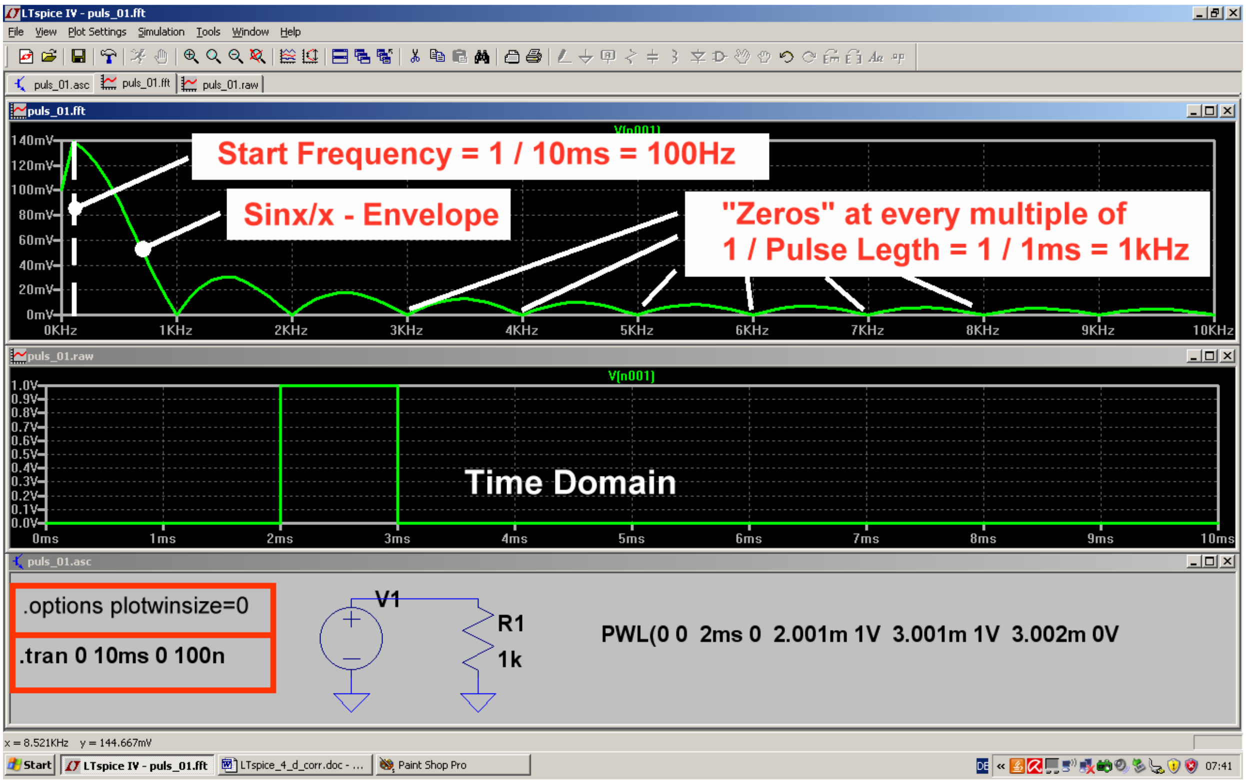
But:
Increasing the number of produced or used number of samples in a FFT gives a higher stop frequency and a higher amplitude dynamic range. In other words: a better resolution (and a longer computation time and a larger data file….)

So right click on the waveform viewer, go to „View“ and „FFT“ and set the sample number to 65536. Start the FFT simulation with OK and afterwards modify the result as follows:

a) Move the cursor to the scaling of the vertical axis and left click: now you can change to „linear presentation“

b) The horizontal axis set to „linear“, using a start frequency of 0Hz, a stop frequency of 10kHz and a tick of 1kHz.

c) Finally choose „tile horizontally“ under „Window“ in the main menu for this result:



Start Frequency = 1 / 10ms = 100Hz

Sinx/x - Envelope

"Zeros" at every multiple of
1 / Pulse Legth = 1 / 1ms = 1kHz

Time Domain

```
.options plotwinsize=0
.tran 0 10ms 0 100n
```

```
PWL(0 0  2ms 0  2.001m 1V  3.001m 1V  3.002m 0V
```

The simulation result confirms the theory which says:

a) In the spectrum of a non-periodic signal you cannot find any discrete frequency lines. The complete frequency range is „loaded“ with energy. So what you see in the diagram is the spectral power density.

b) The amplitude of this density follows to a „**Sin x / x“ envelope function**.

c) “Zeros” in the envelope are always found at every multiple of 1 / pulse length
(Shortening the pulse length increases the “frequency of the fist zero” and all following)

d) The diagram starts with a frequency of 100Hz = 1 / 10ms = 1 / simulation time.

The curve is plotted with thick lines. You find this option by selecting the “button with the hammer” and „Waveforms tab“ as „**Plot data with thick lines**“…

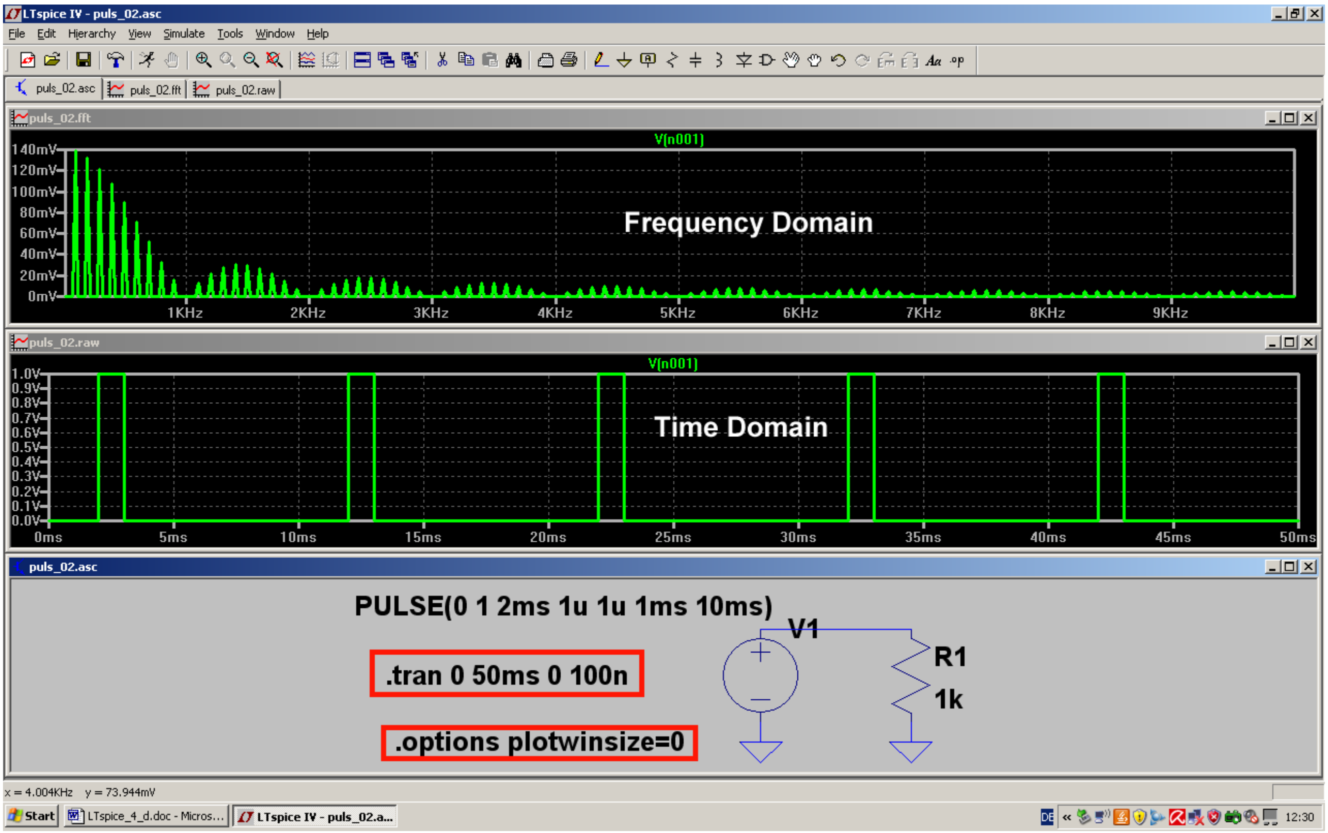
19.3. Simulation of a Periodic Pulse Spectrum

We use the same signal as in the preceding section, but repeat it at 100 times per second. Therefore we need **a pulse voltage source**. In the circuit diagram below you can see how to program the pulse source to achieve this. The simulation time is increased to **50ms** (which gives a frequency resolution of 20Hz) and the maximum time step is set to **100ns**. So we get a minimum of **500 000 actual samples** and can use a number of **262 144 samples for the FFT.**

Now simulate the voltage at R1 and plot the spectrum. Display the FFT result using the following settings:

Vertical axis: **linear**

Horizontal axis: **linear ; start = 0kHz ; tick = 1kHz ; stop = 10kHz**



PULSE(0 1 2ms 1u 1u 1ms 10ms)

.tran 0 50ms 0 100n

.options plotwinsize=0

x = 4.004KHz y = 73.944mV

You can see:

a) It's a **periodic signal,** so we get **a pure „line spectrum**" with the fundamental frequency of 100Hz and the harmonics at every multiple of 100Hz. Due to the simulation time of 50ms the diagram starts at 20Hz. This is also the width of every simulated line.

b) A **pulse length of 1ms gives a Sinx/x envelope function**

c) As before **the "zeros" in the envelope** can be found at

$$\mathbf{F_{NULL} = 1 / pulse\ length = 1 / 1ms = 1kHz}$$

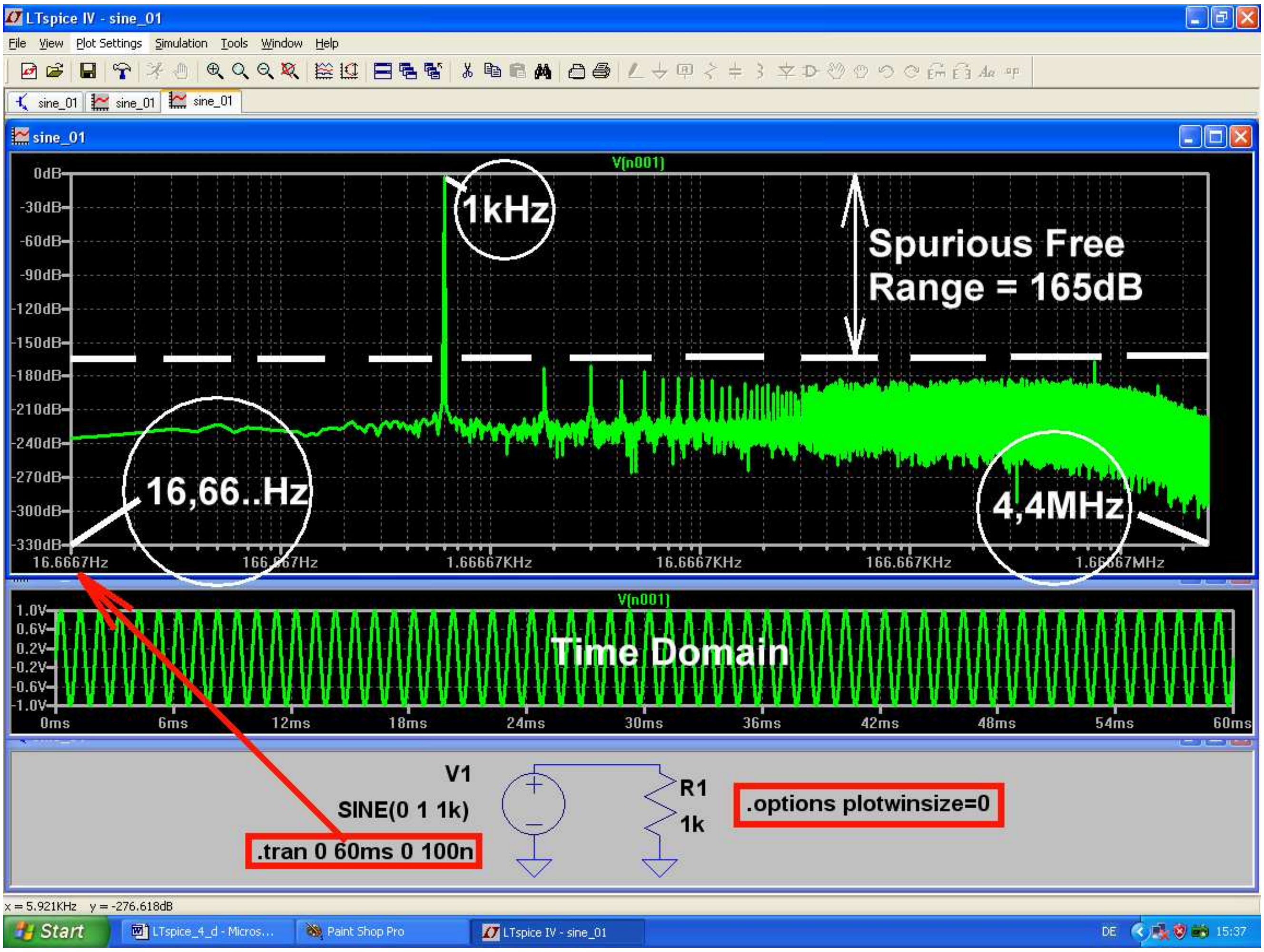
and it's multiples.

19.4. An Ideal Sine Wave

Let us proove the definition which we stated in chapter 19.1 (= fundamentals):

Only a pure continuous sine wave where neither the start point nor the stop point can be observed consists of only one spectral line!

We use a sine wave with a peak value of 1V and a frequency of 1kHz. Simulation time is 60ms using a maximum time step of 100ns. Thus we get 600 000 samples and so it should be possible to find every undesired signal when using 524 288 samples for the FFT. The data compression must also be switched off and the simulation is an integer multiple of the signals's time period (as recommended in chapter 5, FFT)



V1

SINE(0 1 1k)

.tran 0 60ms 0 100n

R1

1k

.options plotwinsize=0

x = 5.921KHz y = -276.618dB

It is true! From 17Hz up to 4.4MHz every undesired signal is attenuated by more but 160dB and only the 1kHz sine wave line can be seen. Nearly ideal and perfect…

19.5. The „Asymmetric Clipped“ Sine Wave

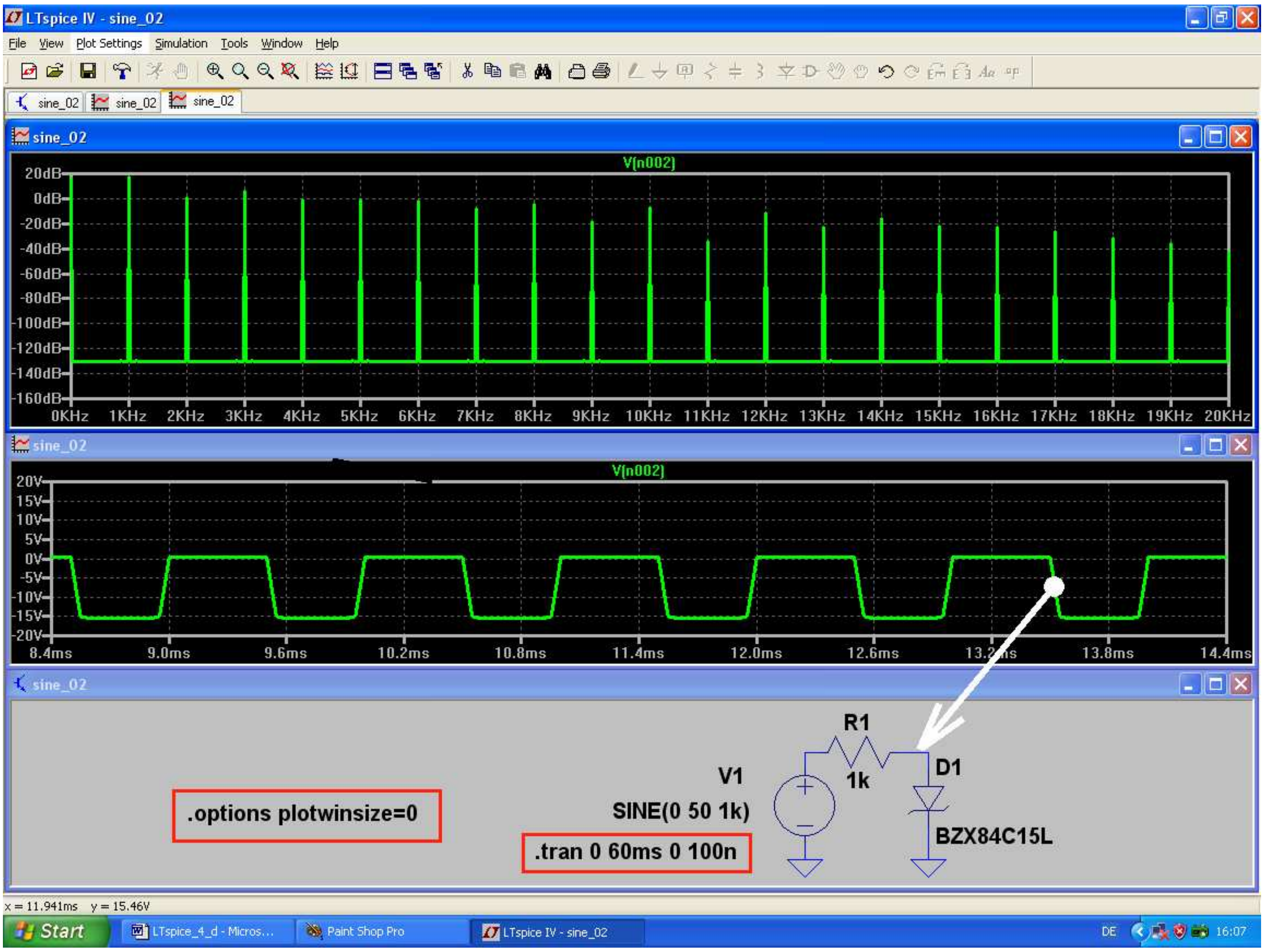
This is done by a zener diode in series with a resistor and showing the simulation of the diode voltage. So the positive half of the sine wave is limited to 0.7V by the conducting silicon diode in the circuit.
When polarity is changing we measure -15V due to the Zener voltage of a BZX84C15L.

But first you have to pick up the diode symbol from the menu and place it. Then right click on the symbol and open the menu „Pick new diode“.

The peak value of the input sine wave must be increased to 50V. Only then the limiting effect can clearly be seen.

Simulation time is 0…60ms with a maximum time Step of 100ns. This gives 60ms/100ns = 600 000 samples for the FFT. Do not forget to switch off the data compression.

On a linear scale show the frequency range from 0 to 20 kHz to see the mixture of even and odd harmonics:



.options plotwinsize=0

SINE(0 50 1k)

.tran 0 60ms 0 100n

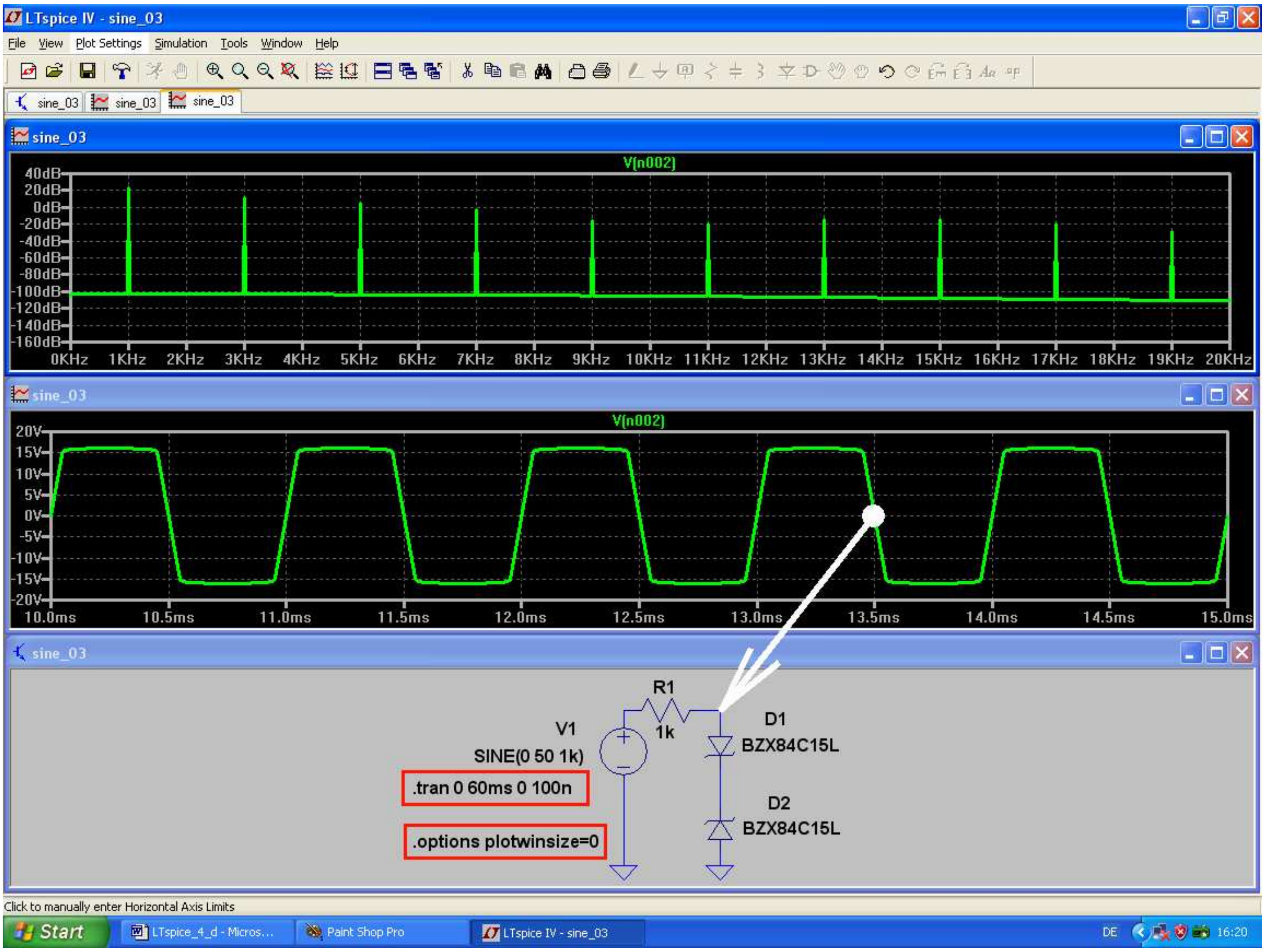
x = 11.941ms   y = 15.46V

19.6. The Symmetrically Clipped Sine Wave

This is a simple exercise because we connect two BZX84C15L zener diodes in series (opposite polarization).

A sine wave is applied again (peak value = 50V, f = 1 kHz). Simulation time is 60ms and the maximum time step is 100ns. This gives 600 000 samples. Data compression is switched off.

We use the same scaling as in section 19.5 and now you see that only odd harmonics can be found.



LTspice IV - sine_03
File View Plot Settings Simulation Tools Window Help
sine_03
V(n002)
40dB
20dB
0dB
-20dB
-40dB
-60dB
-80dB
-100dB
-120dB
-140dB
-160dB
0KHz 1KHz 2KHz 3KHz 4KHz 5KHz 6KHz 7KHz 8KHz 9KHz 10KHz 11KHz 12KHz 13KHz 14KHz 15KHz 16KHz 17KHz 18KHz 19KHz 20KHz
V(n002)
20V
15V
10V
5V
0V
-5V
-10V
-15V
-20V
10.0ms 10.5ms 11.0ms 11.5ms 12.0ms 12.5ms 13.0ms 13.5ms 14.0ms 14.5ms 15.0ms
R1
1k
V1
SINE(0 50 1k)
D1
BZX84C15L
D2
BZX84C15L
Click to manually enter Horizontal Axis Limits
Start
LTspice_4_d - Micros...
Paint Shop Pro
LTspice IV - sine_03
DE
16:20

```
.tran 0 60ms 0 100n
.options plotwinsize=0
```

20. The Secret of the Impulse Response

In modern system theory, remote control and communications the Laplace- and Fourier- and Fast Fourier Transformations are very important tools.

But before starting with an application you have to know the transfer function of the system, which is the ratio of output voltage to the input voltage for all frequencies in complex form (= magnitude and phase).

f you know this "transfer function" then it is possible to predict the "system's response " for any input waveform using two different methods.

==

Method 1 / frequency domain:
Multiply the spectrum of the input signal with the transfer function spectrum and you get the spectrum of the output signal.
Then the inverse Fourier transform will give you the response of the output signal in the time domain.

==

In a complicated system , sometimes the exact calculation or measurement of the transfer function as „magnitude and phase for different frequencies" is difficult or impossible to determine. In these cases use the following:

If the input to an unknown system is a „Dirac - Impulse", then the output of the system has all the information about the transfer function (in the form of the impulse response)!

As we have already seen the power density spectrum of a Dirac-Impulse is constant over the whole frequency range (up to infinity).

This is equal to an AC sweep with a start frequency at 0 Hz and a stop frequency at infinity!

OK, this is a response in the time domain. But time domain and frequency domain simply are the different sides of the same coin! All we then need is a conversion of this response in the time domain to a response in the frequency domain! So we come to

==

Method 2 / time domain:
The output voltage Ua(t) for a behaviour input voltage comes by application of the convolution integral to the input voltage Ue(t) and the impulse response g(t):

$$Ua(t) = Ue(t) * g(t)$$

===

Remarks:
A **Dirac – impulse** is a needle impulse with infinite amplitude and a pulse length decreasing to zero. **But the pulse area is constant and defined as 1Volt x 1Second.**
In practice this is impossible, but it can be approximated by a pulse with huge amplitude and very short pulse length. This pulse length should 100…..1000 times shorter than the system's time constant. If the area of the „single pulse" is the same as that of the Dirac impulse you'll get the same results.

Also in practice you cannot use a Dirac pulse length of 1 Microsecond and an area of 1Volt x 1Second -- this would give a pulse amplitude of 1 Megavolt!

So in an LTI-System (LTI = linear and time invariant) it is customary to apply much smaller pulse amplitudes which can still be handled by the system (e. g. a transistor amplifier). If necessary the answer to a correct pulse area with 1Volt x 1 Second can be calculated by multiplying the measured output signal with the inverse reduction factor.

20.1. First Example: Dirac Pulse applied to a 160Hz RC Lowpass Filter

Task:
Feed a needle pulse with an area of 1Volt x 1Second to the input of a RC LPF **(R = 100kΩ, C = 10nF)**.

a) Calculate the time constant of the LPF and choose a pulse length which is 1000 times shorter.

b) Calculate the pulse amplitude from the pulse length and the pulse area. Use a PWL-source to generate the pulse signal with a **risetime = falltime = 1000 times shorter than the pulse length, a start delay of 1ms and a simulation time from 0 to 10ms with a maximum time step of 100ns. Switch off data compression.**

c) Draw the schematic with the LTspice editor, simulate and present input and output voltage in different plot panes (use „tile horizontally“).

d) Apply an FFT to the input voltage (with **65 536 data samples in time**). Before transforming calculate whether you have enough true (= simulated) samples for a correct FFT.

e) Apply an **FFT** to the output signal.

Use a linear scaling from 0….20kHz with a tick of 1kHz for the frequency axis when presenting the spectrum.

Solution for a): Calculation of the necessary pulse length

We have a corner frequency of

$$f_{CORNER} = \frac{1}{2\pi \bullet RC} = \frac{1}{2\pi \bullet 100k \bullet 10nF} = 159Hz$$

This gives a time constant of:

$$RC = \frac{1}{2\pi \bullet f_{CORNER}} = \tau = 100k\Omega \bullet 10nF = 1ms$$

So we use a pulse length which is 1000 times shorter = **1µs.**

Solution for b) and c): Simulation using the PWL source

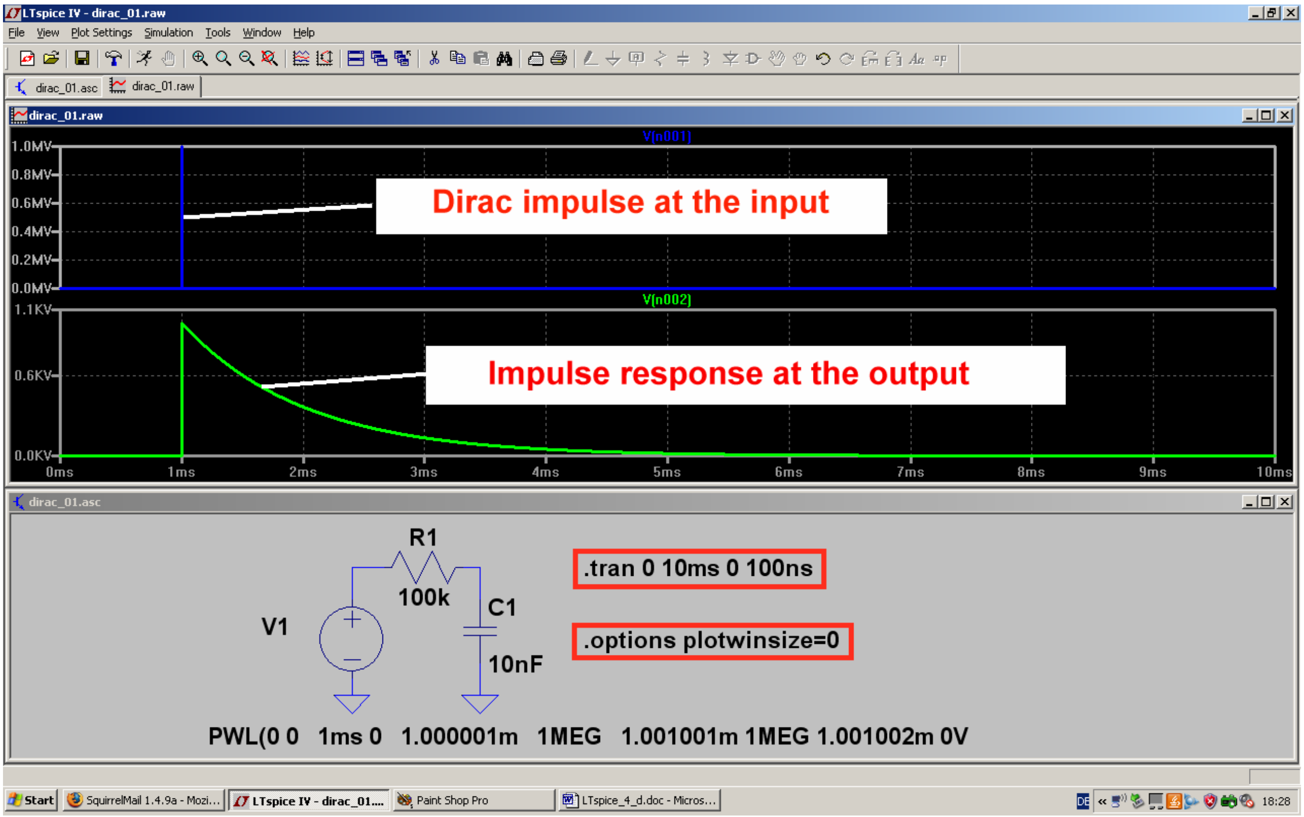
The pulse length is 1µs, **so the rise time and the fall time must each be 1ns (= 1µs / 1000).**

A pulse length of 1µs and a pulse area of 1 Volt x 1 second gives a pulse amplitude of
1 Megavolt = (1V x 1s) / 1µs.

The PWL source is now programmed as follows:

At T = 0:	**Amplitude = 0**
At T = 1ms	**Amplitude = 0**
At T = 1.000001ms	**Amplitude = 1MEG**
At T = 1.001001ms	**Amplitude = 1MEG**
At T = 1.001002ms	**Amplitude = 0**

See the simulation result screen on the next page including the properties of the PWL source.


LTspice IV - dirac_01.raw
File View Plot Settings Simulation Tools Window Help
dirac_01.asc
dirac_01.raw
V(n001)
1.0MV
0.8MV
0.6MV
0.4MV
0.2MV
0.0MV
V(n002)
1.1KV
0.6KV
0.0KV
0ms
1ms
2ms
3ms
4ms
5ms
6ms
7ms
8ms
9ms
10ms
R1
100k
C1
10nF
V1
Start
SquirrelMail 1.4.9a - Mozi...
LTspice IV - dirac_01....
Paint Shop Pro
LTspice_4_d.doc - Micros...
18:28

Dirac impulse at the input

Impulse response at the output

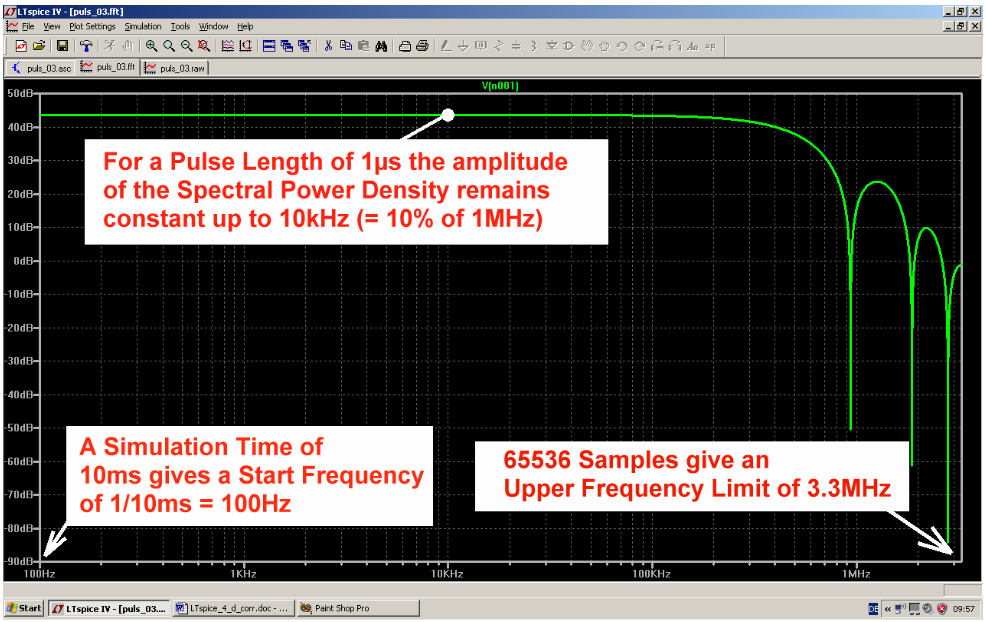
.tran 0 10ms 0 100ns

.options plotwinsize=0

PWL(0 0 1ms 0 1.000001m 1MEG 1.001001m 1MEG 1.001002m 0V)

Solution for d): FFT of the input signal

Simulating for 10ms with a maximum time step of 100ns gives a minimum of 10ms / 100ns = 100 000 true samples. So right click on the trace of the input voltage in the diagram, choose "View" and "FFT" and use **65536 sampled data points.** Close with OK and examine:



For a Pulse Length of 1µs the amplitude of the Spectral Power Density remains constant up to 10kHz (= 10% of 1MHz)

A Simulation Time of 10ms gives a Start Frequency of 1/10ms = 100Hz

65536 Samples give an Upper Frequency Limit of 3.3MHz

a) **start frequency**

=

frequency resolution

=

minimum frequency step

=

minimum spectral line width

which are determined by the simulation time of 10ms = 1 / 10ms = 100Hz.

b) The upper frequency limit in the diagram (= stop frequency) is determined by the number of samples used for the FFT **AND** by the maximum time step (which gives the minimum sample frequency in the time domain).

But:

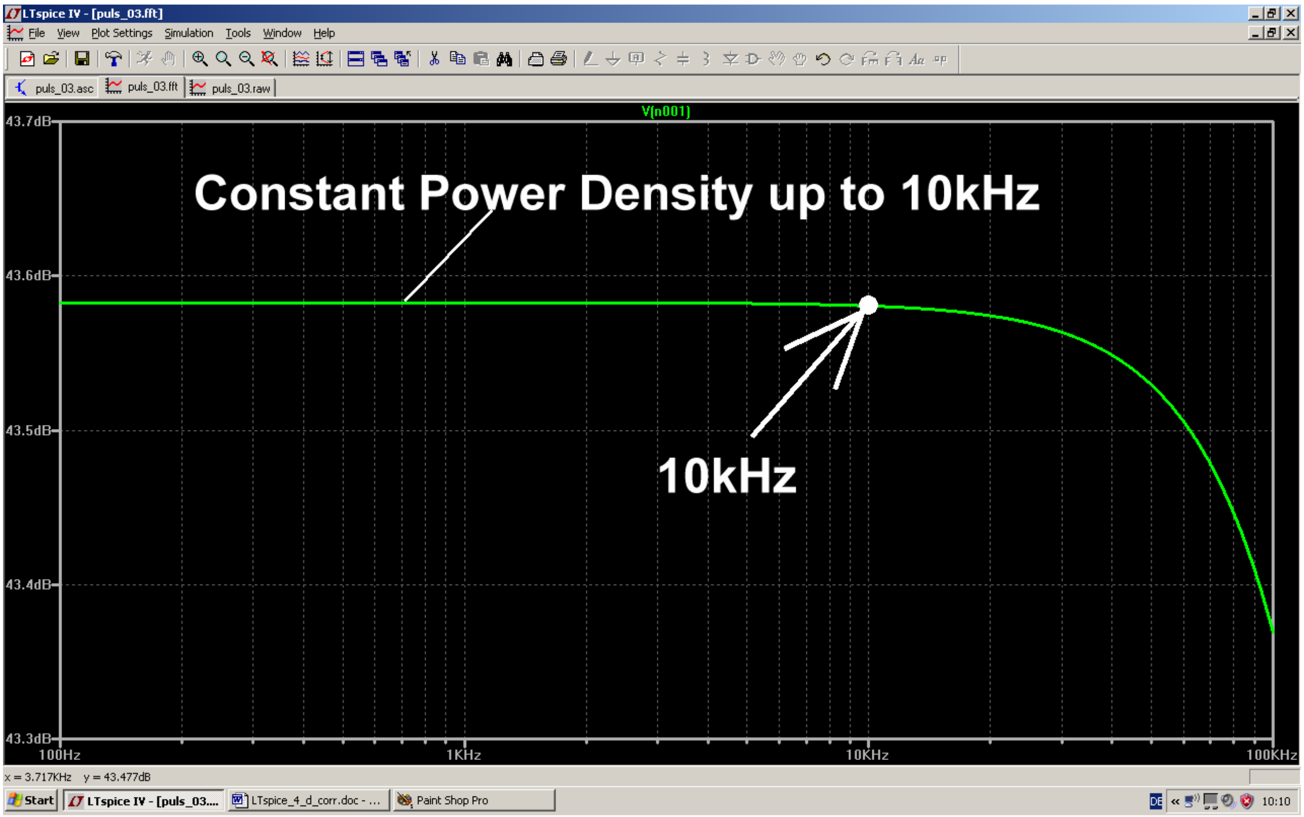
For a correct FFT you always need a “multiple of 2” samples (e. g. 128 or 256 or 512 or 1024 or 2048….). Do not try to change this. Always check by calculation whether you have enough “true sampled values” before calling the FFT. This number is simply given by the ratio “**simulation time / minimum time step”.**

c) A pulse length of 1µs gives a first “zero“ in the spectrum’s envelope at

1 / 1µs = 1MHz

but the Sinx/x function of the envelope only shows a constant amplitude of the spectral power density up to 10% of the „first ZERO frequency”. For this example this means 10% of 1MHz = 10kHz.

This is equal to an AC sweep with constant input voltage amplitude up to a stop frequency of 10kHz. Check this by zooming the diagram for a frequency range from 100Hz to 100kHz and an expanded amplitude range:


LTspice IV - [puls_03.fft]
File View Plot Settings Simulation Tools Window Help
puls_03.asc
puls_03.fft
puls_03.raw
V(n001)
43.7dB
43.6dB
43.5dB
43.4dB
43.3dB
100Hz
1KHz
10KHz
100KHz
Constant Power Density up to 10kHz
10kHz
x = 3.717KHz   y = 43.477dB
Start
LTspice IV - [puls_03....
LTspice_4_d_corr.doc - ...
Paint Shop Pro
DE
10:10

Additional Tasks:

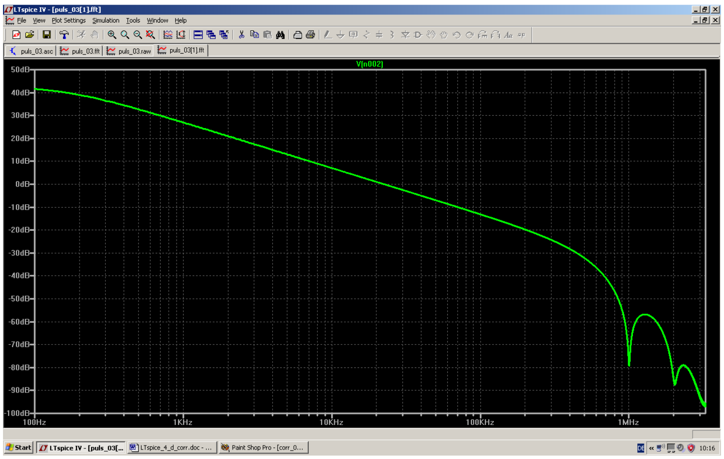
Repeat the simulation and the FFT with different simulation times (e. g. 20 ms / 50ms / 100ms) and compare with a).

Repeat the FFT with different numbers of sampled data points and compare (e. g. 16384 / 32768 / 131072 / 262144 samples).

Solution for e): FFT of the impulse response = output signal
Remember:

Time domain and frequency domain are simply the two different sides of the same coin! Both completely describe the properties of our LPF.

So when applying an FFT to the output signal, we use „convolution“ which would then show the transfer function of our LPF as a result.

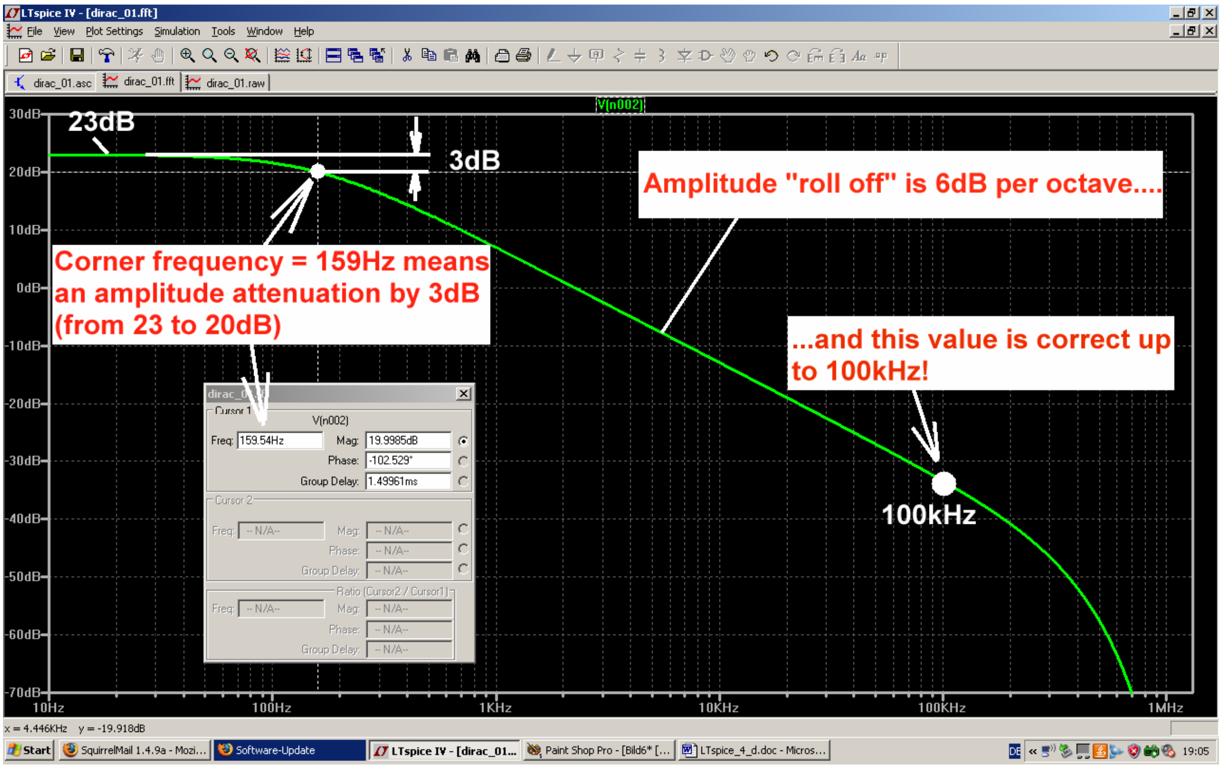


At higher frequencies the influence of the “single Dirac impulse” can be seen. This means that we see the Sinx / x – envelope with zeros belonging to the NON-ZERO pulse length. So do not use the results in this region.

But the results at the low frequencies should also be improved. The corner frequency of the LPF is 159Hz and so a frequency resolution of 10Hz should be applied.

Now repeat the simulation with a stop time of 1 / 10Hz = 100ms. The minimum time step can be reduced to 200ns without loosing to much information or decreasing the „number of samples“ to less than 500 000 (check: 100ms / 200ns = 500 000).

When using 262 144 samples you see everything what you want:



Amplitude "roll off" is 6dB per octave....

Corner frequency = 159Hz means an amplitude attenuation by 3dB (from 23 to 20dB)

...and this value is correct up to 100kHz!

Cursor 1 V(n002)
Freq: 159.54Hz Mag: 19.9985dB
Phase: -102.529°
Group Delay: 1.49961ms

Cursor 2
Freq: --N/A-- Mag: --N/A--
Phase: --N/A--
Group Delay: --N/A--

Ratio (Cursor2 / Cursor1)
Freq: --N/A-- Mag: --N/A--
Phase: --N/A--
Group Delay: --N/A--

x = 4.446KHz y = -19.918dB

Start frequency AND frequency resolution are now exactly 10Hz.

At 159 Hz we find the corner frequency of the passband. Here the amplitude is exactly reduced by 3dB.

In the stop band the amplitude “rolls off” (= decreases) with 6dB per octave.

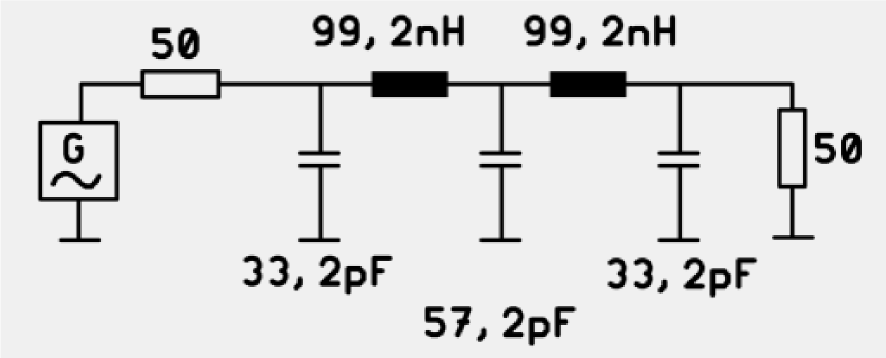
This roll off value of 6dB per octave is constant and correct up to nearly 100kHz.
If you want to improve this: repeat the simulation with a shorter pulse length. This shifts the first “null” in the envelope function to higher frequencies.

20.2. Second Example: Dirac Test of a 110MHz Lowpass Filter (see Chapter 14.2)

20.2.1. Simulating S21 (= Forward Transmission)

These were the specifications of chapter 14.2

„Ripple“ corner Frequency	**fc = 110 MHz**
Minimum inductance version	
Filter order	**n = 5**
System resistance	**Z = 50Ω**
„Passband ripple“	**0,1 dB.**



…and these were the part values calculated by a LC-filter design program.

Now the task:

a) Start an AC sweep of the circuit and **plot S21 from 100kHz to 1000MHz with 401 points per decade**. Zoom the result to demonstrate the “Tchebychev ripple“ of 0.1dB in the passband.

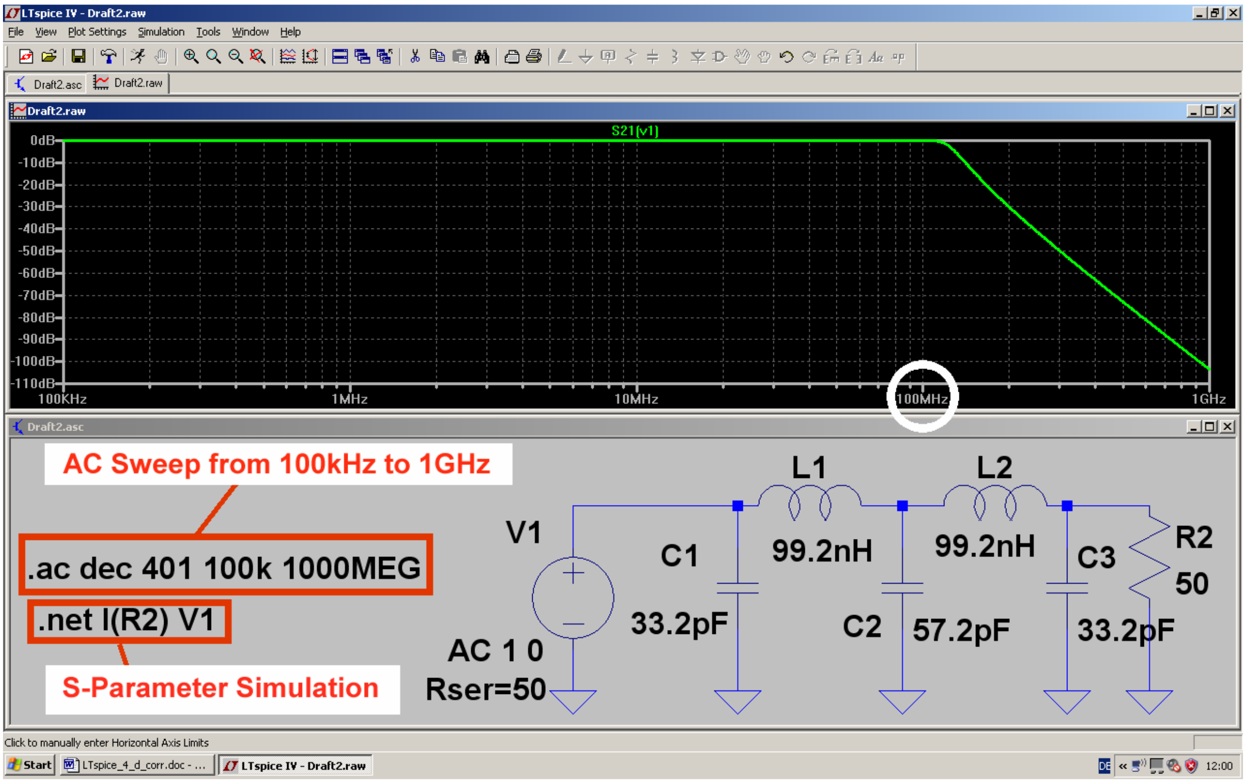
b) Replace the swept input voltage by a „ Dirac impulse“ and use a pulse length which is 100 times shorter than the system’s time constant.
Rise time and fall time should be 100 times shorter than the pulse length.
The pulse amplitude is 1 Megavolt. A start frequency = frequency resolution of 100 kHz is required.
Calculate the necessary sweep time and the maximum time step of the transient simulation for 100 000 true samples (and use 65536 samples when starting the FFT).
The simulation starts after a delay of **1ns**.

c) Choose the Maximum Time Step so short that the programm cannot oversee the Dirac impulse (e.g. 2 or 2 or 3 three samples during the Dirac impule length and simulate in the Time Domain. Switch the data compression off.

d) Calculate the number of usable samples for the FFT and apply the FFT to the impulse response of the filter. Show the Chebychev ripple with it’s amplitude of 0.1dB in the passband and compare the result to the S parameter sweep.

e) Repeat the FFT with only **65 536 samples** and compare the result to d).

Solution for a): Simulated S21 from 100kHz to 1GHz (phase plot switched off):



AC Sweep from 100kHz to 1GHz

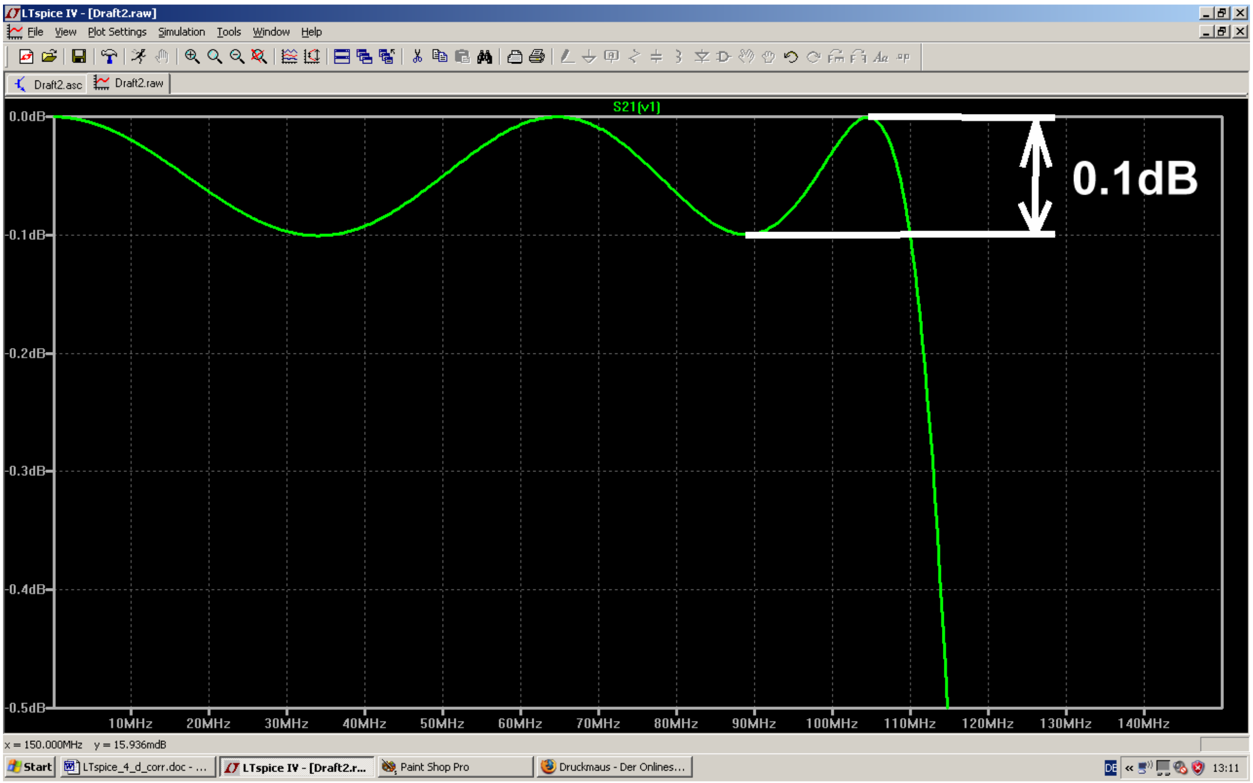
.ac dec 401 100k 1000MEG

.net I(R2) V1

S-Parameter Simulation

Click to manually enter Horizontal Axis Limits

….and this is the passband ripple of 0.1dB:



x = 150.000MHz y = 15.936mdB

This picture is the „specimen“ for all the following work.

Solution for b) and c):
The corner frequency of the LPF is 110MHz. This gives a time constant of

$$\tau = \frac{1}{2\pi \bullet f_{CORNER}} = \frac{1}{2\pi \bullet 110MHz} = 1.44ns$$

So the length of the Dirac impulse must be 100 times shorter. We choose ca. **15 picoseconds.**

Rise time and fall times are reduced to (1 / 100 of the pulse length) = **0.15ps = 0.00015ns.**

The amplitude of the pulse is limited to 1Kilovolt. Otherwise we will get problems with the computer's numeric precision capability for this extremely short pulse length. This is customary for an LTI-System like this LPF.

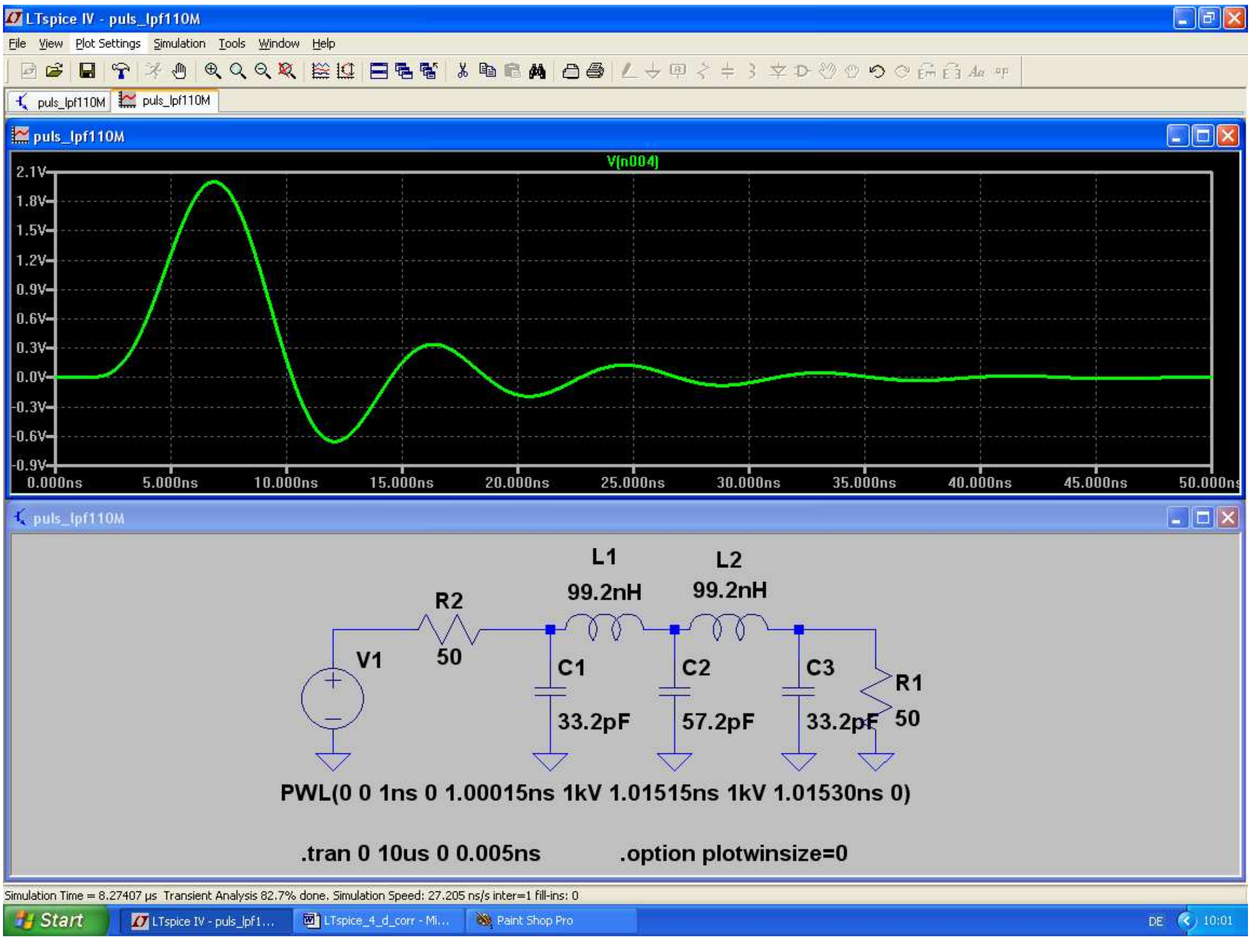
Together with the start delay of 1ns and the amplitude maximum of 1 MegaVolt we get the following properties for the PWL source:

At T = 0:	**U = 0**
At T = 1ns:	**U = 0**
At T = 1,00015ns:	**U = 1 kiloVolt (= 1kV)**
At T = 1,01515ns:	**U = 1 kiloVolt (= 1kV)**
At T = 1,01530ns:	**U = 0**

For a start frequency = frequency resolution = 100kHz in the simulated spectrum a transient simulation time of

$$t_{max} = 1 / 100kHz \ = 10\mu s$$

is necessary. And a Maximum Time Step of **15 picoseconds / 3 = 5 picoseconds** is a good choice that the Dirac impulse will not be overseen by SPICE. Data compression is switched to OFF.



PWL(0 0 1ns 0 1.00015ns 1kV 1.01515ns 1kV 1.01530ns 0)

.tran 0 10us 0 0.005ns .option plotwinsize=0

Simulation Time = 8.27407 µs Transient Analysis 82.7% done. Simulation Speed: 27.205 ns/s inter=1 fill-ins: 0

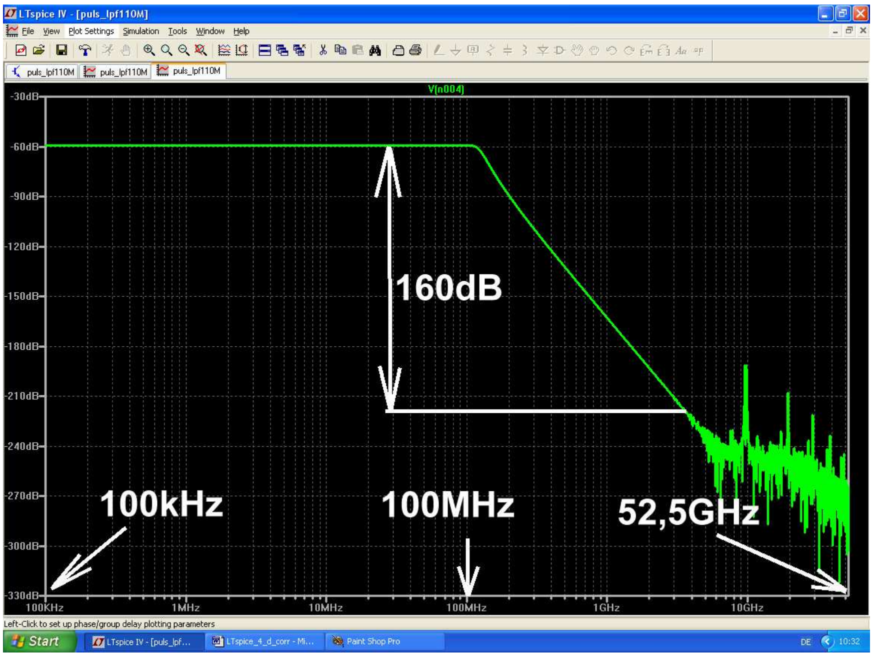
These are the first 50ns of the output voltage = **impulse response.**

Solution for d): FFT

A simulation time of 10µs and a Maximum Time Step of 5 picoseconds give

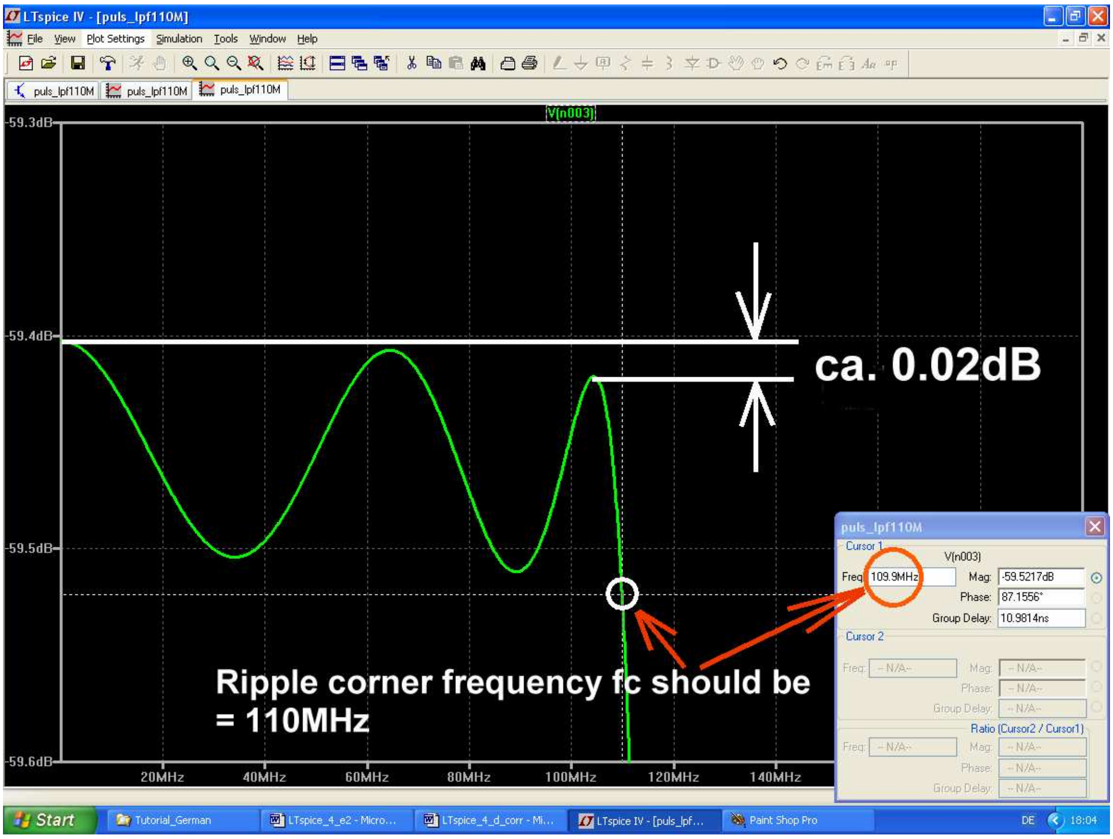
10µs / 0.005ps = 2 000 000 samples

This means that we can use 1 048 576 values for the FFT.



This is the simulation result for a linear frequency axis:

And when zooming we can see the Chebychev ripple and compare to the S parameter sweep:



Ripple corner frequency fc should be = 110MHz

puls_lpf110M

Cursor 1

V(n003)

Freq: 109.9MHz Mag: -59.5217dB ●

Phase: 87.1556° ○

Group Delay: 10.9814ns ○

Cursor 2

Freq: --N/A-- Mag: --N/A-- ○

Phase: --N/A-- ○

Group Delay: --N/A-- ○

Ratio (Cursor2 / Cursor1)

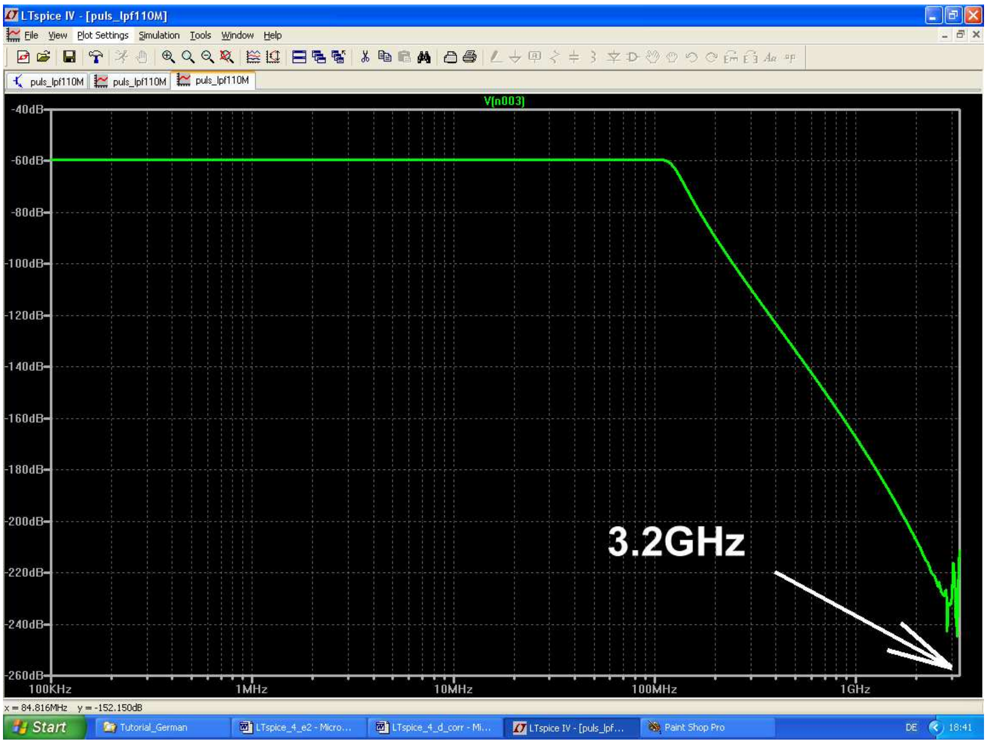
Freq: --N/A-- Mag: --N/A--

Phase: --N/A--

Group Delay: --N/A--

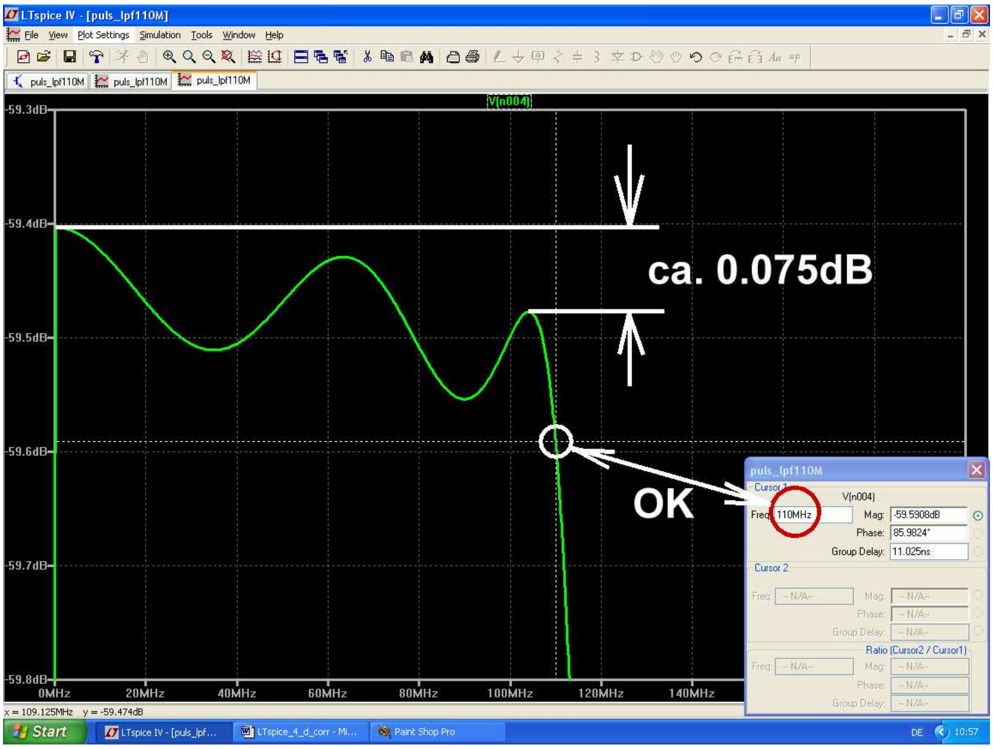
Not so bad….

Solution for e): simulation with only 65536 samples



```
x = 84.816MHz   y = -152.150dB
```

Reducing the number of samples for the FFT will also reduce the simulation accuracy AND the the maximum frequency presented in the diagram from 52.5GHz to 3.2GHz



puls_lpf110M

Cursor 1

V(n004)

Freq: 110MHz   Mag: -59.5908dB

Phase: 85.9824°

Group Delay: 11.025ns

Cursor 2

Freq: --N/A--   Mag: --N/A--

Phase: --N/A--

Group Delay: --N/A--

Ratio (Cursor2 / Cursor1)

Freq: --N/A--   Mag: --N/A--

Phase: --N/A--

Group Delay: --N/A--

```
x = 109.125MHz   y = -59.474dB
```

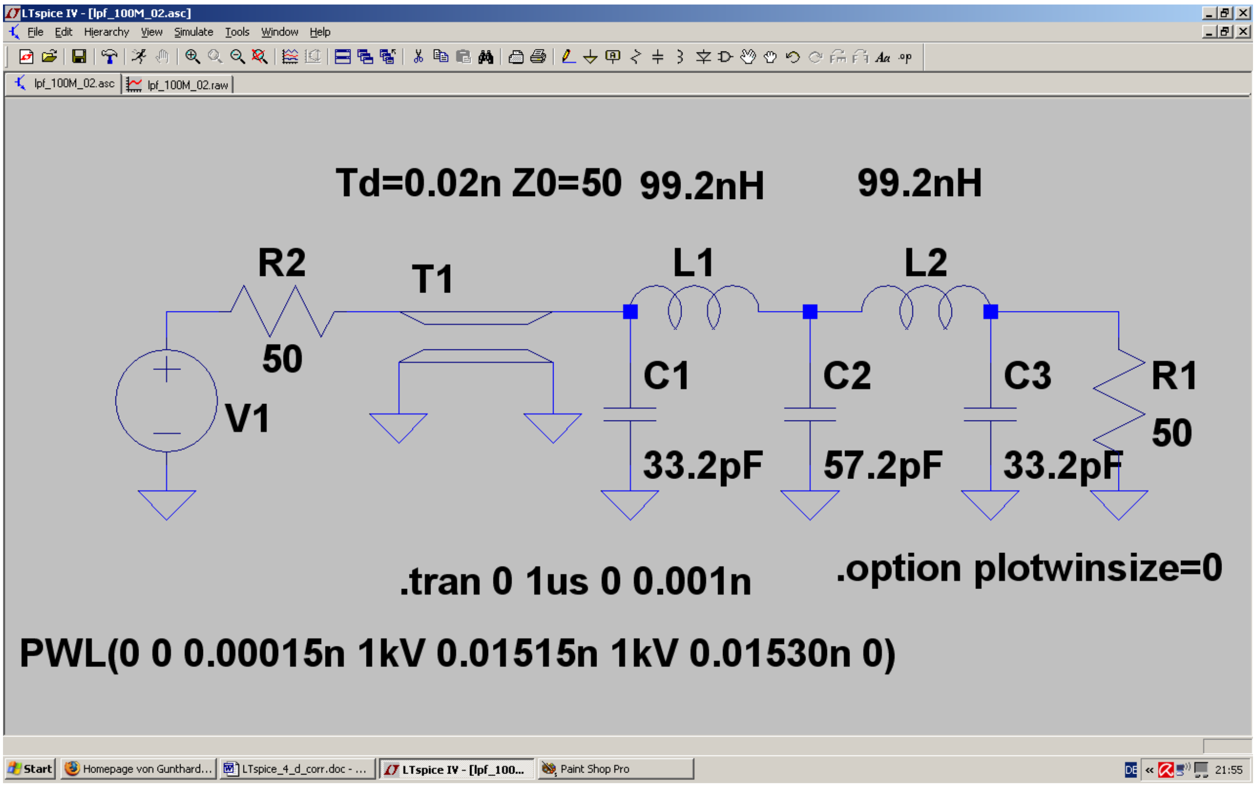
The ripple corner frequency value with 110MHz is correctly simulated, but there is an error of 0.075B at the Chebychev ripple calculation.

20.2.2. Simulation of S11 (= input reflection) or S22 (= output reflection)

The S parameters of a Twoport consist of two transmission coefficients (S21 and S12) and two reflection coefficients (S11 and S22). In practice for the determination of the reflection coefficients directional couplers are used to separate the incident and the reflected wave.

But in a simulation this is very easy when using a Dirac impulse: we insert a short piece of RF cable between the voltage source and the port. The travelling time for the Dirac impulse on this piece of cable should be greater than the pulse length. Now an echo caused by mismatch on the port's input will arrive after 2 x (travelling time on the cable) at the cable's entry. This echo can be used for the FFT and to determine S11 or S22 in dB.

This is the used schematic to determine S11:



```
.tran 0 1us 0 0.001n
.option plotwinsize=0
PWL(0 0 0.00015n 1kV 0.01515n 1kV 0.01530n 0)
```

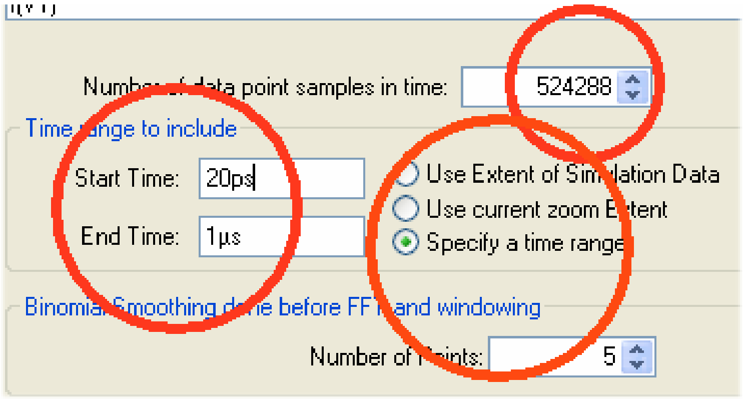
Voltage source V1 is connected to the input of a 50Ω cable with a signal travelling time of **20 picoseconds (= part "tline" in LTspice)**. The output of this cable feeds the input of the LPF with the Dirac impulse (**amplitude = 1 kiloVolt, pulse length = 15 picoseconds, rise and fall time = 0.15 picoseconds, start at t = 0)**. The following value pairs are used to generate this impulse:

At t = Null:	**U = Null**
At t = 0,00015ns:	**U = 1 kiloVolt**
At t = 0,01515ns:	**U = 1 kiloVolt**
At t = 0,01530ns:	**U = Null**

So the echo can be seen and measured at t = 40 picoseconds at cable's entry. **But for the calculation of the FFT this time slot from 0….40 picoseconds must be suppressed.**

Data compression is switched OFF by **.option plotwinsize=0**.

To reduce the PC calculation time only a **simulation time of 1 µs** is used. This gives a frequency resolution of **1 / 1µs = 1MHz** which will do for a S11 calculation. But the Maximum Time Step was reduced to 1 picosecond. So for a pulse length of 15 picoseconds for the input Dirac pulse (…and an echo pulse length in the same order) enough details of the curve are sampled.
This gives **1µs / 0.001ns = 1 000 000** calculated samples of which 524288 are used for the FFT.



Number of data point samples in time: 524288

Time range to include

Start Time: 20ps

End Time: 1µs

○ Use Extent of Simulation Data

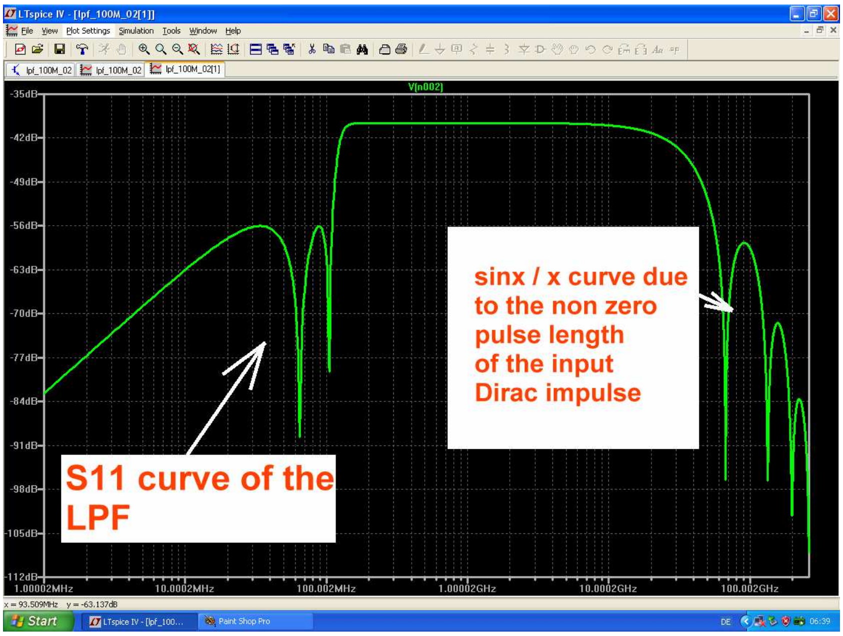
○ Use current zoom Extent

● Specify a time range

Binomial Smoothing done before FFT and windowing

Number of Points: 5

Here come the necessary entries for the FFT. **Do not forget to ignore the time range from 0….20 ps** with the input pulse and use 524288 samples.

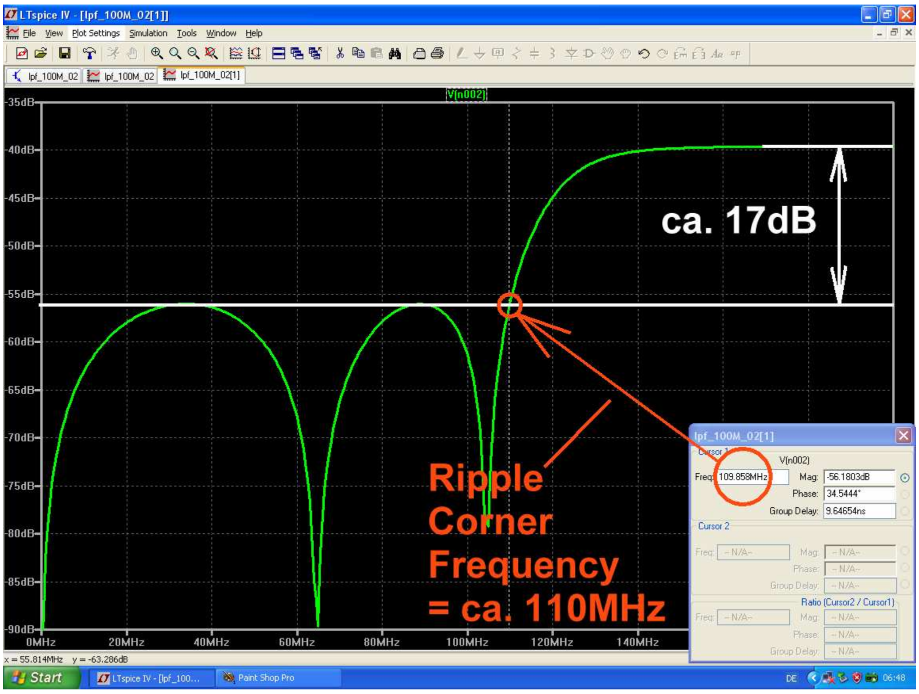


sinx / x curve due to the non zero pulse length of the input Dirac impulse

S11 curve of the LPF

This is the result: the S11 curve of the LPF appears at the left hand side of the diagram. At the right hand side the sinx / x curve for the “spare Dirac pulse” due to its non-zero pulse length is present at high frequencies.

But: the maximum frequency value at the horizontal axis is 250 GHz enabled by the short Maximum Time Step of 1 picosecond.



ca. 17dB

Ripple Corner Frequency = ca. 110MHz

lpf_100M_02[1]

Cursor 1

V(n002)

Freq: 109.858MHz Mag: -56.1803dB

Phase: 34.5444°

Group Delay: 9.64654ns

Cursor 2

Freq: --N/A-- Mag: --N/A--

Phase: --N/A--

Group Delay: --N/A--

Ratio (Cursor2 / Cursor1)

Freq: --N/A-- Mag: --N/A--

Phase: --N/A--

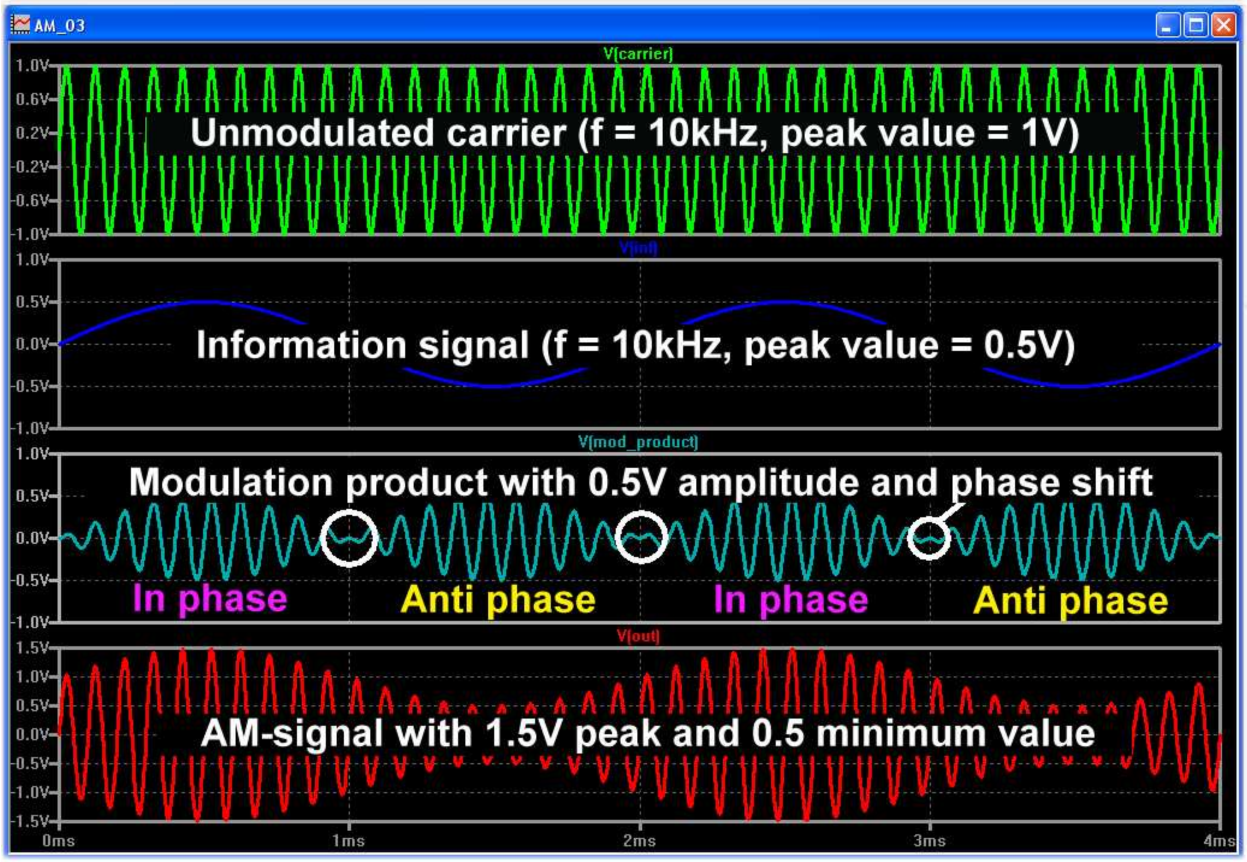
Group Delay: --N/A--

When zooming the result everything is OK and equates to the S parameter sweep in chapter 14.2.

21. Modulation

21.1. Principle of Amplitude Modulation (AM)

It is probably the oldest type of modulation that enabled the transmission of language and music approximately 100 years ago following on from Wireless Telegraphy using a Morse key. The basic idea is simple: Audio signals have wavelengths that are far too large to be radiated wirelessly using antennas with acceptable efficiency. Therefore a high frequency carrier signal was used so that a smaller antenna could be used and the information loaded onto the carrier for problem free transmission. The amplitude of the carrier is changed in sympathy with the information. Mathematically this is a multiplication and addition of two different signals. The secret of this is shown in the following figure:



The top trace shows the constant 10kHz carrier with a peak value of 1V. Below that is the information that is a 500Hz sine wave with a peak value of 0.5V. If these two signals are multiplied the result is a signal at the carrier frequency where:

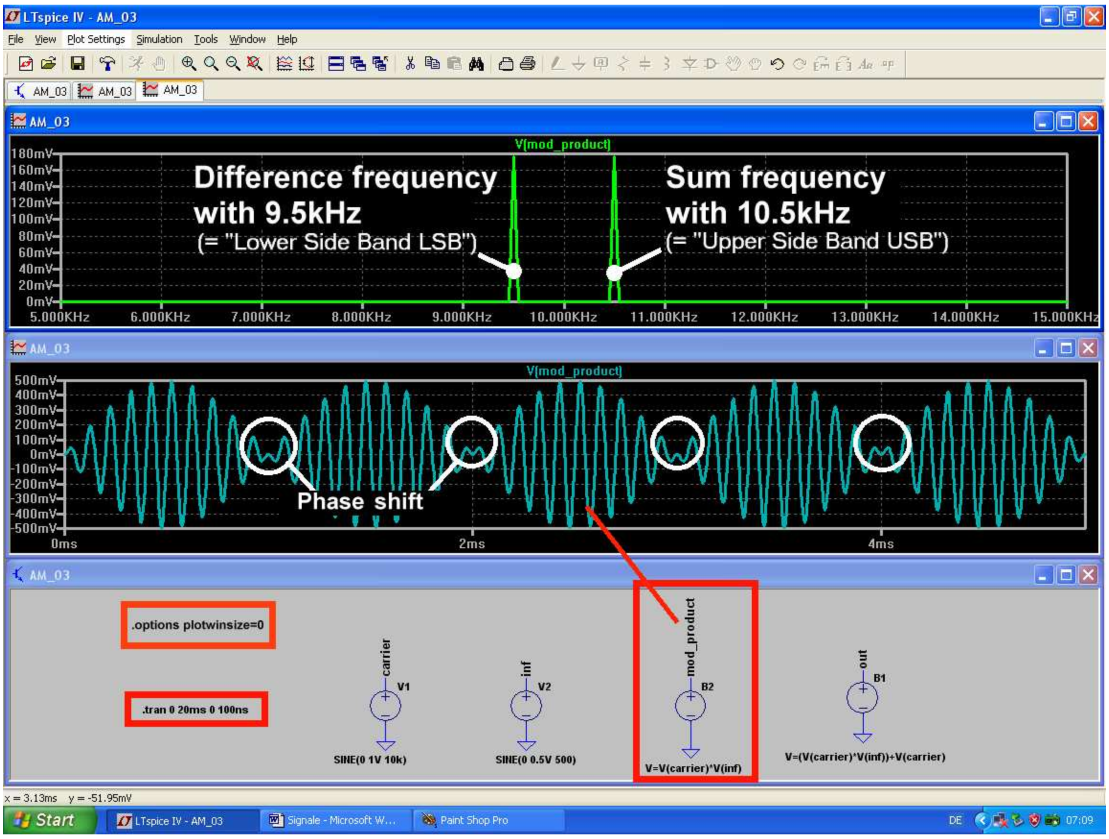
- The amplitude follows the information signal but:

- When the information signal voltage goes negative the phase reverses.

This multiplication is called the Modulation Product shown in the third trace. If this is added to the unmodulated carrier signal the total voltage becomes larger if the modulation product is in phase with the carrier. Likewise it becomes smaller if the modulation product and carrier are out of phase. The bottom trace is the AM signal where the signal amplitude varies in sympathy with the information. Now the question is: how many different frequencies are involved because this is no longer a constant sine wave? With the carrier signal it is simple because it is a constant sine wave with only one spectrum line. The modulation product already has a mathematical formula for the multiplication of two sine signals:

$$\cos(\alpha)\bullet\cos(\beta)=\frac{1}{2}[\cos(\alpha+\beta)+\cos(\alpha-\beta)]$$

The products are the sum and difference frequencies of the two cosine signals. The carrier and information frequencies are not present.



Difference frequency with 9.5kHz (= "Lower Side Band LSB")

Sum frequency with 10.5kHz (= "Upper Side Band USB")

Phase shift

.options plotwinsize=0

.tran 0 20ms 0 100ns

SINE(0 1V 10k)

SINE(0 0.5V 500)

V=V(carrier)*V(inf)

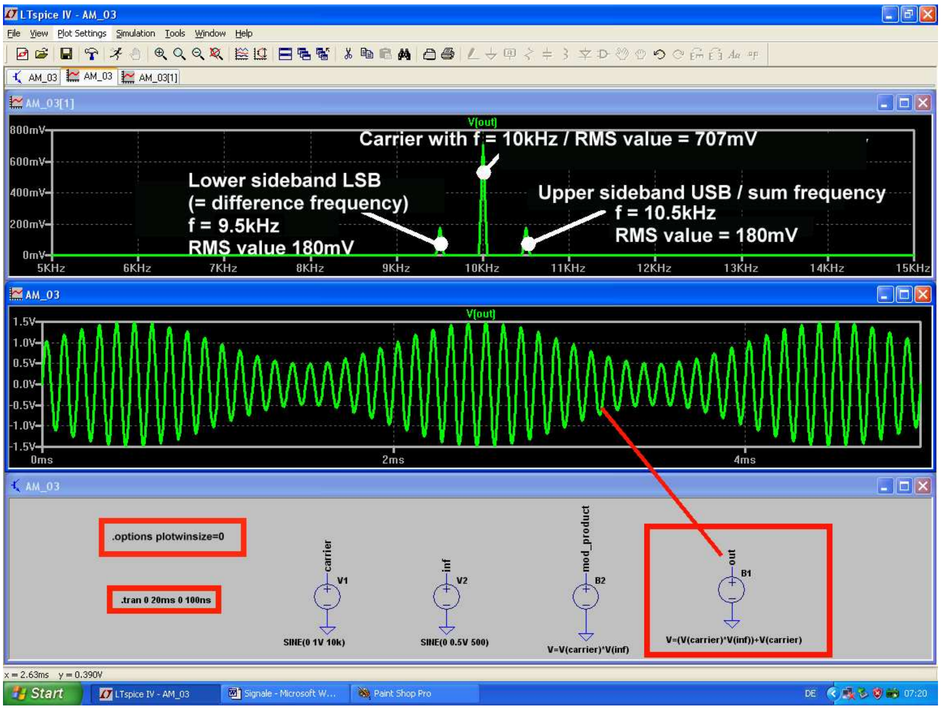
V=(V(carrier)*V(inf))+V(carrier)

x = 3.13ms y = -51.95mV

To simulate this we use the Behavioural Voltage (bv), this has the necessary tools to simulate the multiplication of the two signals. This is confirmed in the above figure:

- The sum frequency is 10kHz + 0.5kHz = 10.5kHz
- The difference frequency is 10kHz - 0.5kHz = 9.5kHz

They are called upper and lower sidebands having equal amplitudes. The peak value of both signals amounts to 50% of the information signal thus 250mVs. In the spectrum display this is 180mV because of the RMS representation.



Carrier with f = 10kHz / RMS value = 707mV

Lower sideband LSB (= difference frequency) f = 9.5kHz RMS value 180mV

Upper sideband USB / sum frequency f = 10.5kHz RMS value = 180mV

.options plotwinsize=0

.tran 0 20ms 0 100ns

SINE(0 1V 10k)

SINE(0 0.5V 500)

V=V(carrier)*V(inf)

V=(V(carrier)*V(inf))+V(carrier)

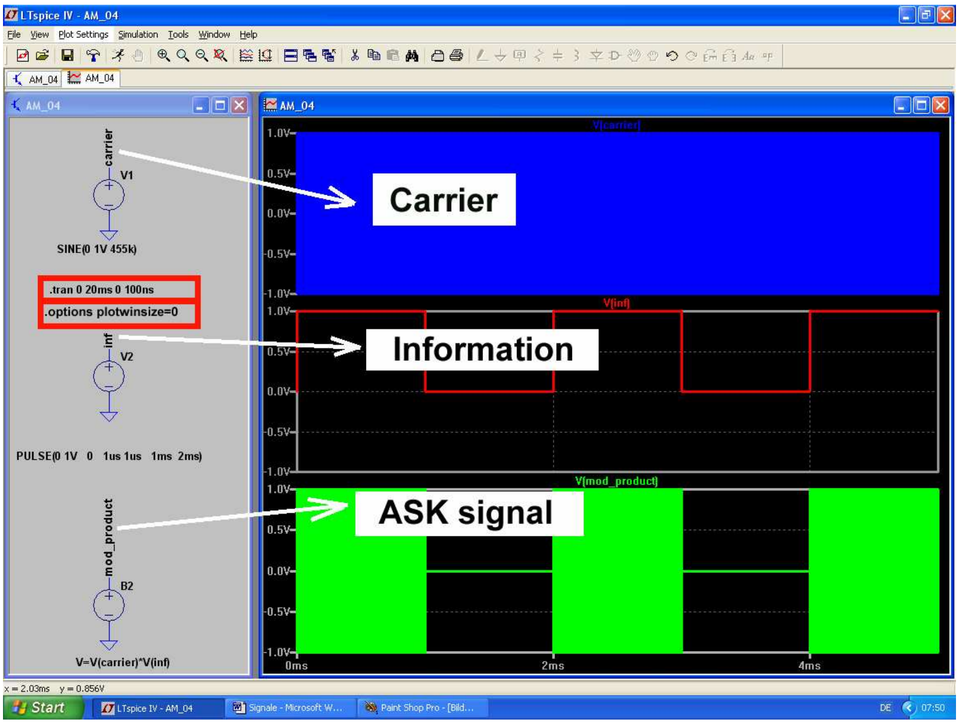
x = 2.63ms y = 0.390V

For the complete AM signal the carrier signal is added to the modulation product. This can be simulated using the bv source as shown in this figure.

Note:
Usually several frequencies or a complete frequency band are involved in the information signal. These are called the lower and upper sidebands, the Lower Side Band is the LSB and the Upper Side Band is the USB of the AM signal.

21.2 Amplitude Shift Keying (ASK)

The good old Morse key gave just “Open” and “Closed”. This simple method still has its place today in as a modern transmission technique, the logical levels are simply assigned to the two conditions “Zero” and “One“. This works amazingly well and is reasonably interference-proof (if additional limiting is used). In addition the circuits required are fairly simple. The bv source is used again for the simulation multiplying the carrier with a 1kHz symmetrical square wave information signal. The smallest voltage level is zero V, resulting from “Key open”. The maximum voltage is 1V resulting from “Key closed”.



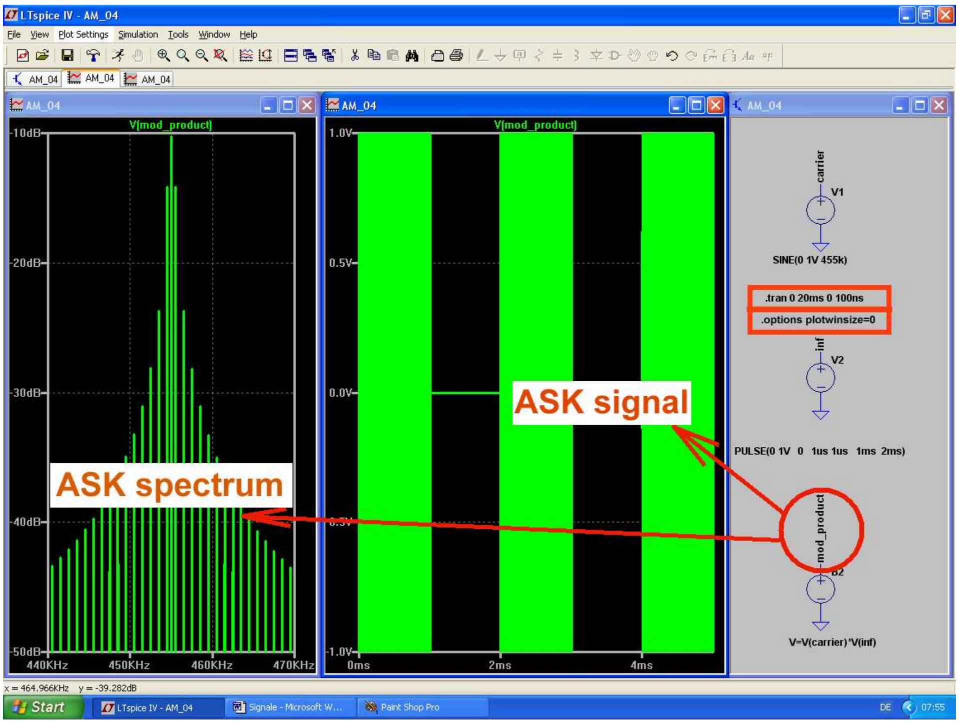
SINE(0 1V 455k)

.tran 0 20ms 0 100ns
.options plotwinsize=0

PULSE(0 1V 0 1us 1us 1ms 2ms)

V=V(carrier)*V(inf)

A carrier signal of 455kHz was selected for this example. This figure shows the simulation circuit with the output signals of the three voltage sources used with a peak value of 1V or 0.707V RMS.



SINE(0 1V 455k)

.tran 0 20ms 0 100ns
.options plotwinsize=0

PULSE(0 1V 0 1us 1us 1ms 2ms)

V=V(carrier)*V(inf)

The spectrum of the ASK signal is shown in Fig 22, it shows what was to be expected.
The symmetrical square wave information signal contains the 1KHz fundamental and all odd number harmonics. The result is LSBs and USBs with all these spectral lines of the information signal that spread to the left and right of the carrier.

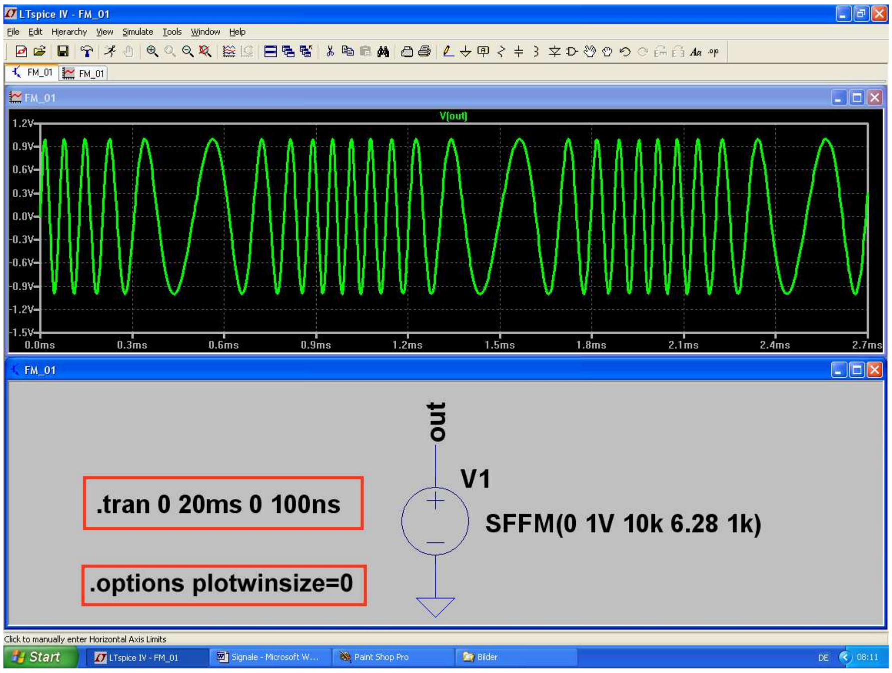
21.3 Frequency modulation (FM)

This is a refinement because this time the amplitude of the carrier is not used. Yes, it is completely unimportant and in an FM receiver a limiter removes all fluctuations and glitches. The carrier frequency is changed in the sympathy with the information signal. That costs more and clearly requires substantially higher technical expertise. Success is convincing, it can be heard by listening to good music received on a VHF receiver. It is good that LTspice has three components available:

• One can be used to switch the voltage supply to produce FM (SFFM). A sinusoidal carrier is produced and sinusoidal information signal is automatically superimposed.

• There is complete FM-AM generator with a “modulate” function that can be found in the “Special Functions” section. It is suitable for information signals with different waveforms. • Finally the same FM-AM generator has a “modulate2” function that produces quadrature outputs (two separate outputs with a 90 degree phase shift between the carrier signals).

21.3.1. Generating an FM signal using the SFFM voltage supply



.tran 0 20ms 0 100ns

.options plotwinsize=0

SFFM(0 1V 10k 6.28 1k)

Take a close look at this figure. The frequency of the signal increases during the negative half wave and decreases during the positive half wave of the information signal. Unfortunately the modulating information signal cannot be shown on a separate diagram in this mode. Data compression is switch off as usual. It is simulated over 20ms (100ns resolution) but for clarity only a short section is shown. The rest can be seen from the SFFM source programming. The line:

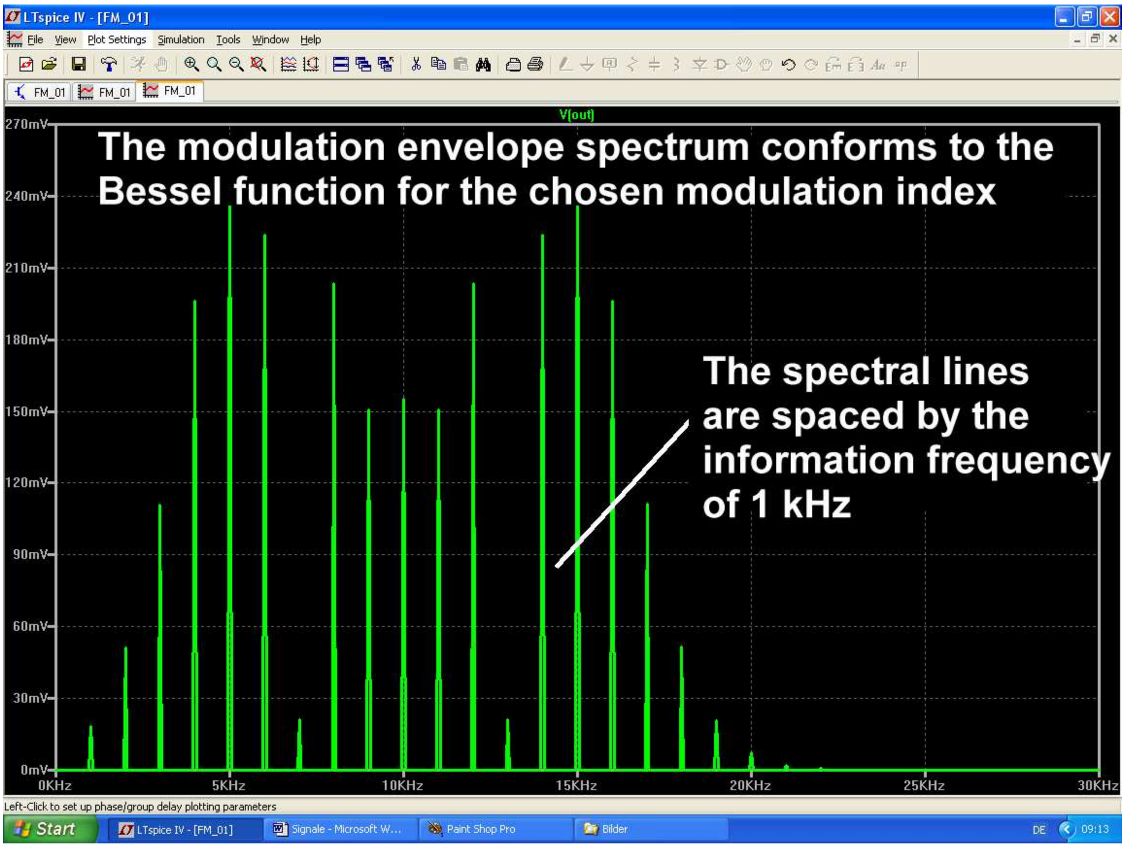
SFFM (0 1V 10k 6.28 1K)

means:

• 0	**= no DC portion**
• 1V	**= peak value of the carrier**
• 10k	**= 10kHz unmodulated carrier frequency**
• 6.28	**= modulation index**
• 1k	**= 1kHz sinusoidal information signal**

The modulation index needs an explanation. If the carrier frequency increases it can be interpreted as advancing the phase in relation to the unmodulated signal. Similarly a lower frequency means a reduction of the phase in relation to the unmodulated signal. The modulation index expresses this change of the phase. It represents the maximum phase shift in relation to the steady state with the phase deviation measured in radians. For the example the modulation index of 6.28 represents the phase shift of a circle with a radius of 1. A complete circle is 360 degrees or 6.28 radians. It is so simple!

Now let us regard the simulated FFT spectrum of this signal using 131,072 samples.



The modulation envelope spectrum conforms to the Bessel function for the chosen modulation index

The spectral lines are spaced by the information frequency of 1 kHz

Left-Click to set up phase/group delay plotting parameters

The entire spectrum is occupied with lines spaced at a distance of the information signal frequency. The amplitude distribution envelope can be described using a Bessel functions. It is difficult to understand the detail of this but simply the frequency range occupied by the FM signal increases with the modulation index but the envelope and the number of spectral lines obey the Bessel functions. In an extreme case increasing the modulation index, the unmodulated carrier would reach zero.

The **frequency deviation** indicates the maximum deviation of the instantaneous frequency in relation to the unmodulated signal and this naturally related to the modulation index (this is sometimes known a the phase change). As an example the VHF broadcast stations from 88 to 108MHz use a frequency deviation of ±75kHz. That seems small but it cannot be more otherwise the maximum permissible channel spacing (300kHz) would be exceeded and the transmitter on the adjacent channel would be disturbed.

As a guide:

FM channel spacing = 2 x (frequency deviation + information signal frequency)

For interest the phase change can be calculated from the frequency deviation and the information signal frequency and the modulation index in radians can be calculated using:

Modulation index = (peak frequency deviation) / (information frequency)

The differences between frequency modulation and phase modulation are:

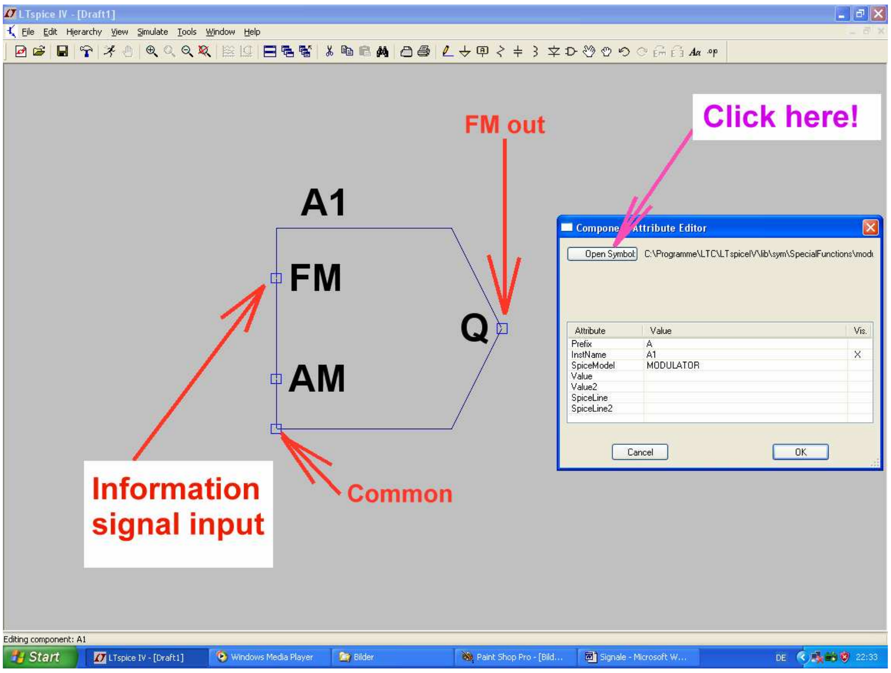
• For **frequency modulation** the **peak frequency deviation is kept constant**. From the formula the modulation index and thus the maximum phase fluctuation must decrease with rising information frequency. Unfortunately this worsens the signal signal-to-noise ratio for high tones.

• For **phase modulation** the **modulation index and phase fluctuation are kept constant**. According to the formula the frequency deviation increases with increasing information signal frequency. This increases the frequency range occupied.

21.3.2. Frequency Shift Keying (FSK)

Frequency Shift Keying is a digital mode that assigns the two conditions, “logic 0” and “logic 1” to two different frequencies. The origin of this mode comes from the days of teleprinter technology. The only things that remain from that technology are the two names, **“Mark”** for the higher frequency and “**Space”** for the lower frequency. These are required to programme the **FM-AM generator “modulate” function of the LTspice simulator**.

Unfortunately some work is needed. When the VCO (Voltage Controlled Oscillator) is selected the basic adjustment is missing. In order to correct this, click the right mouse button on the circuit symbol and open the “Component Attributes Editor”. Click on the “Open Symbol” button to change to the actual symbol editor (recognised by the small red circles at the ends of the individual lines that emerge in the circuit symbol and represent connection points).



Component Attribute Editor

Open Symbol: C:\Programme\LTC\LTspiceIV\lib\sym\SpecialFunctions\modu

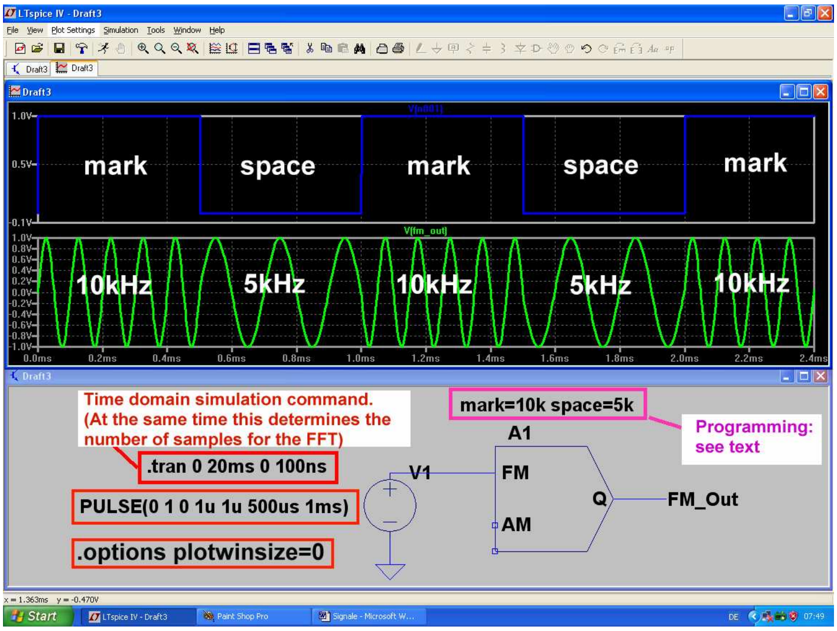
Attribute	Value	Vis.
Prefix	A	
InstName	A1	X
SpiceModel	MODULATOR	
Value		
Value2		
SpiceLine		
SpiceLine2		

Cancel OK

The path “**Edit/attributes/Edit At**tributes” leads to the “**Symbol Attribute Editor**” as shown in the figure. In the line “Value” enter “**mark=10k**” and “**space=5k**” as the control characteristic of the VCO to give:

- **For 0V at the FM input the output frequency will be the space value of 5kHz.**
- **For 1V at the FM input the output frequency will be the mark value of 10kHz.**

For a beginner this entry is somewhat complicated: First click on the line “Value” and then type “mark=10k space=5k” in the window above the table. Click OK to make the entry - however it is not shown on the screen. Now select the path: “**Edit/Attributes/Attributes Window**” again and click on “Value” in the table then click OK and the cursor changes to “Mark Space Programming” so that it can be can placed on the circuit symbol.
Now this changed symbol must be stored in the component library with a path like:
“LTspiceIV / lib / sym/ Special Functions / modulate”.



mark=10k space=5k

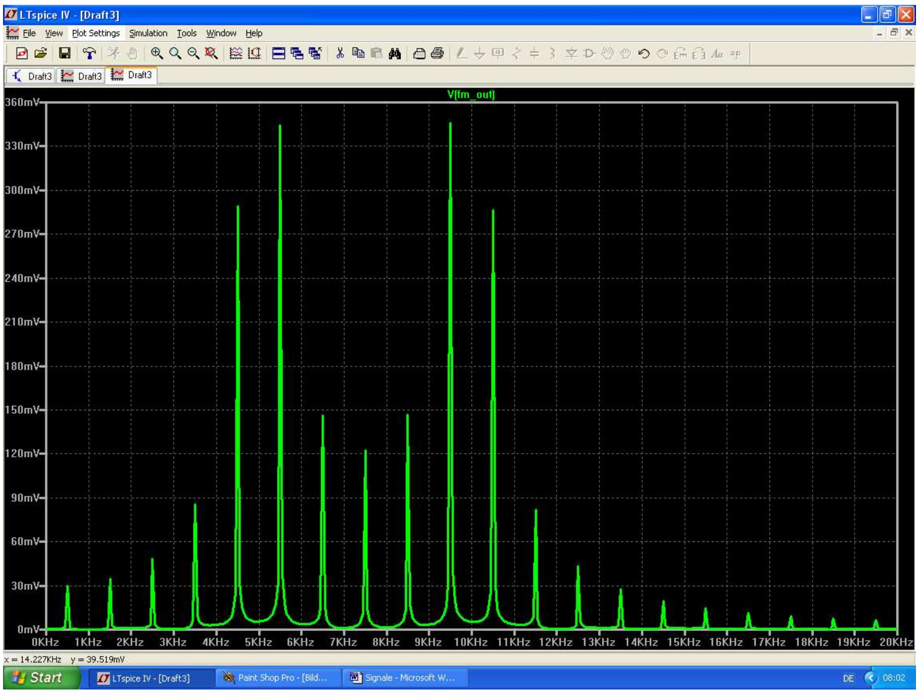
.tran 0 20ms 0 100ns

PULSE(0 1 0 1u 1u 500us 1ms)

.options plotwinsize=0

Now this VCO will be available but only if all windows are closed and the program restarted. Otherwise the program does not know anything about the changed symbol.
The figure shows the symbol being used with a 1kHz pulse with **Vmin = 0V and Vmax = 1V connected to the FM input** and the simulated output (FM_Out).

To ensure that the additionally voltage information is shown, right mouse click on the diagram and use the “Add Plot Pane/Add trace”. For the following FFT simulation enter the attributes “20ms simulation time” and “maximum time step 100ns” in order to receive 200,000 genuine samples. Also no data compression should be used.


LTspice IV - [Draft3]
File View Plot Settings Simulation Tools Window Help
Draft3 Draft3 Draft3
V(fm_out)
360mV
330mV
300mV
270mV
240mV
210mV
180mV
150mV
120mV
90mV
60mV
30mV
0mV
0KHz 1KHz 2KHz 3KHz 4KHz 5KHz 6KHz 7KHz 8KHz 9KHz 10KHz 11KHz 12KHz 13KHz 14KHz 15KHz 16KHz 17KHz 18KHz 19KHz 20KHz
Start
LTspice IV - [Draft3]
Paint Shop Pro - [Bild...
Signale - Microsoft W...
DE
08:02

x = 14.227KHz y = 39.519mV

The linear frequency spectrum simulated with 131,072 samples is shown here.

It exhibits two maxima, one for “Mark” and one for “Space”. This is a wellknown fact that can be found in technical literature.

It occupies a considerable frequency range, probably 20kHz.

For interested readers who have access to a computer there are some additional exercises:

- Change the minimum and maximum pulse amplitude at the input. Examine if negative voltage levels are permissible.

- Use a 1kHz sine wave information signal in place of the pulse. Select the amplitude similar to “Mark” and “Space”. Also use an offset for the sine wave. Compare the frequency range now occupied to the pulse simulation example.

- Use triangle information signal with the same frequency of 1kHz.