DIGITAL WE DAYS 2023

LTSPICE FOR THE VALIDATION OF POWER CONVERTERS

WÜRTH ELEKTRONIK MORE THAN YOU EXPECT

TODAY'S SPEAKERS

PRESENTATION
Sylvain Le Bras
Field Application Engineer

MODERATION
Markus Eberle
Marketing Department

INFORMATION ABOUT THE WEBINAR

You are muted during the webinar.
However, you can ask us questions using the chat function.

Duration of the presentation	30 Min
Q&A:	10 – 15 Min

Any questions?
No problem! Email us digital-we-days@we-online.com

Please help us to optimize our webinars!
We are looking forward to your feedback.

On our channel	Würth Elektronik Group
And on	Digital WE Days 2023 YouTube Playlist

USAGE OF LTSPICE SIMULATION TO SPEED UP CONVERTER DESIGN

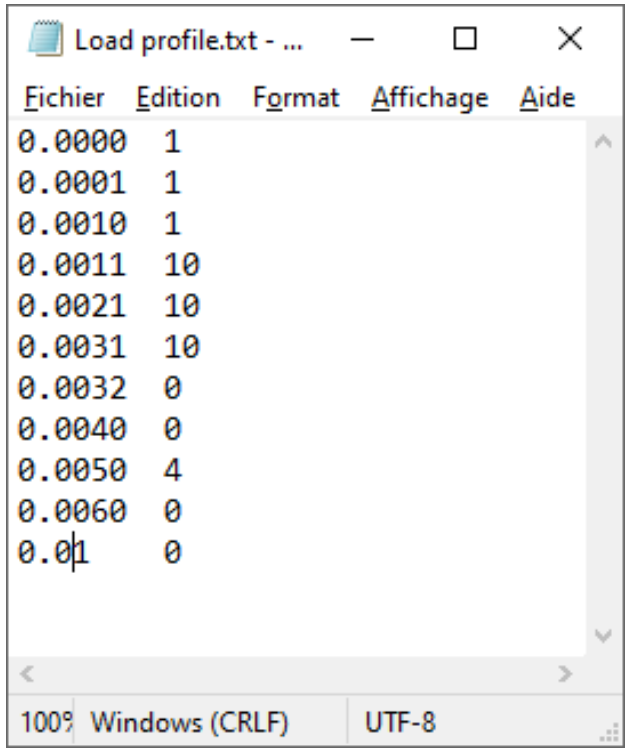
Letting the computer work for you is a science

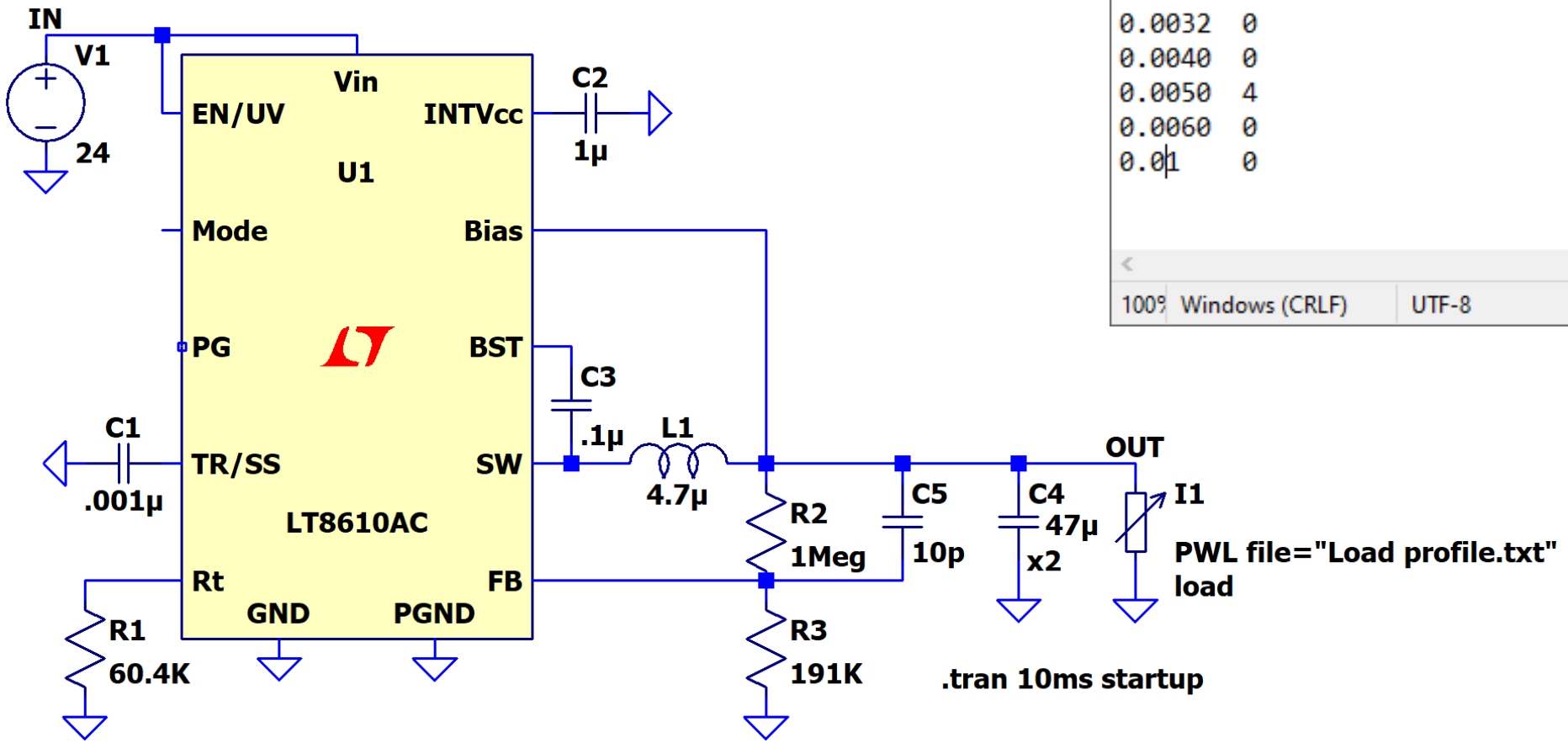
- Usage of PWL Test vectors
  - Load profiles
  - ~~Transient injection~~
- Iteration of simulation
  - Parameters and simulation Steps
  - Monte Carlo, Worst Case and Gaussian distribution
- Measure statement
  - Basic usage
  - Advanced usage (computing power factor)
  - Plot Meas'd data
- « Exotic » plot coordinates and their usage
  - Safe Operating Area of a mosfet (simple)
  - ~~Safe Operating Area of a mosfet (pulse time weighted)~~
  - Thermal transient « SOATHERM »

USAGE OF PWL TEST VECTORS

Generation, acquisition, and use of PWL vectors

- Converters are stressed by variations of the output load
- Load transient simulation cuts down validation time

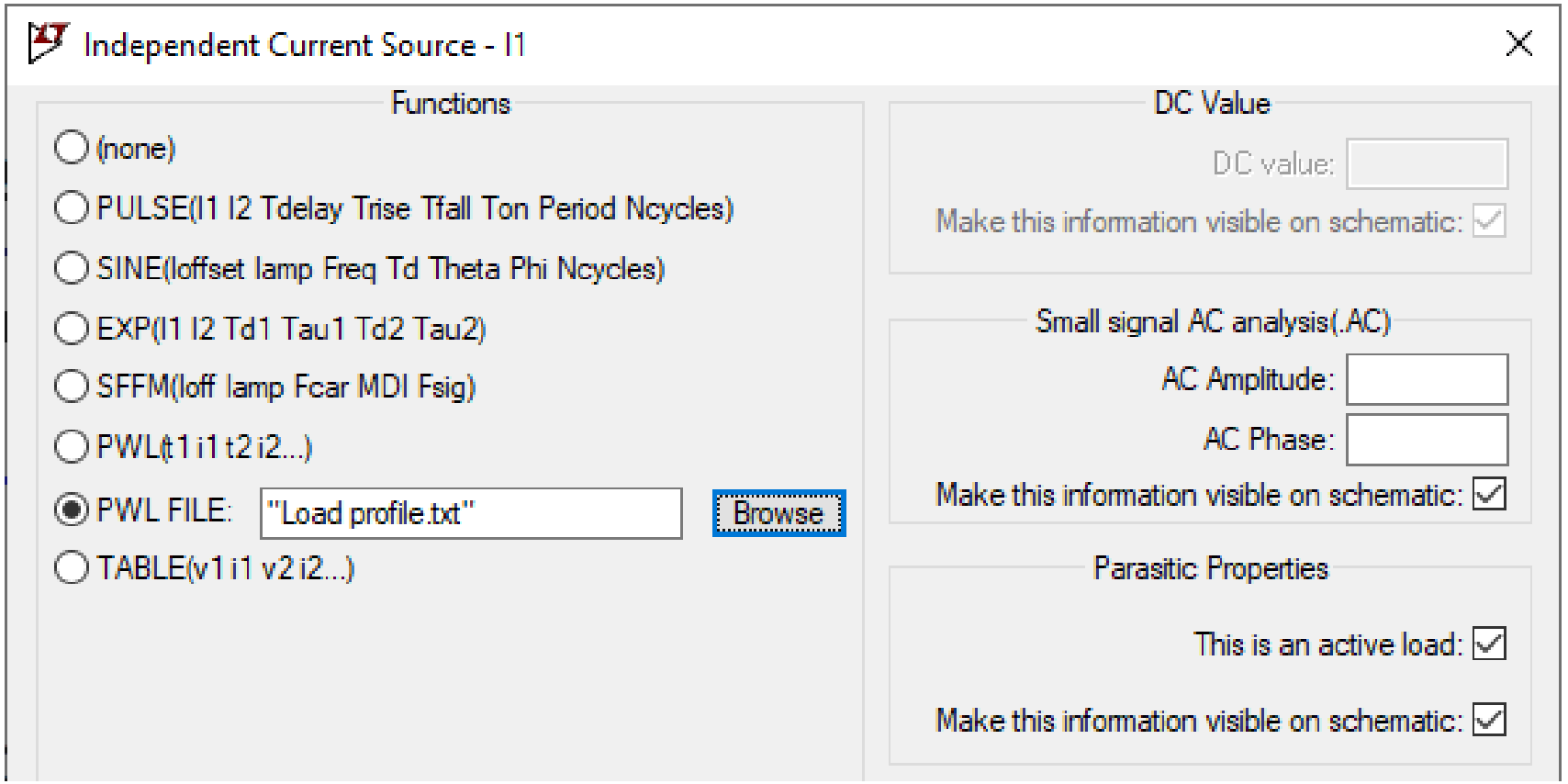


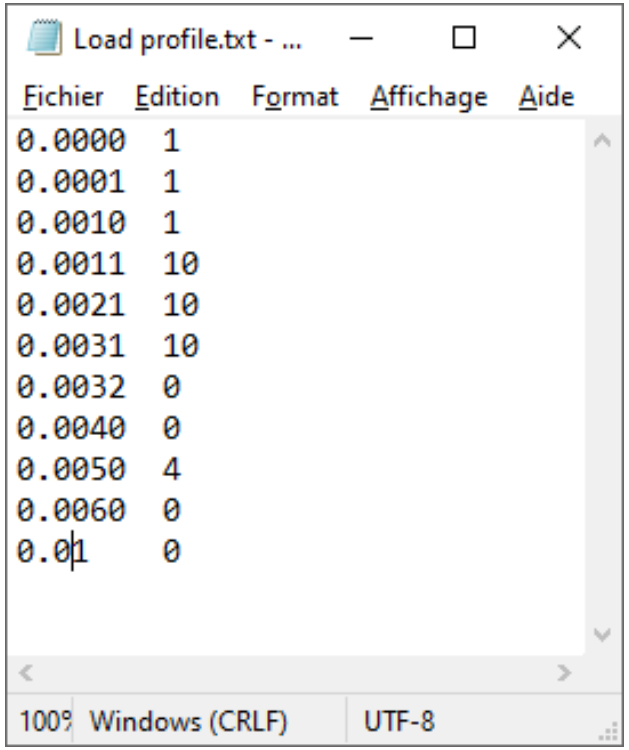


USAGE OF PWL TEST VECTORS

Generation, acquisition, and use of PWL vectors

- Converters are stressed by variations of the output load
- Load transient simulation cuts down validation time

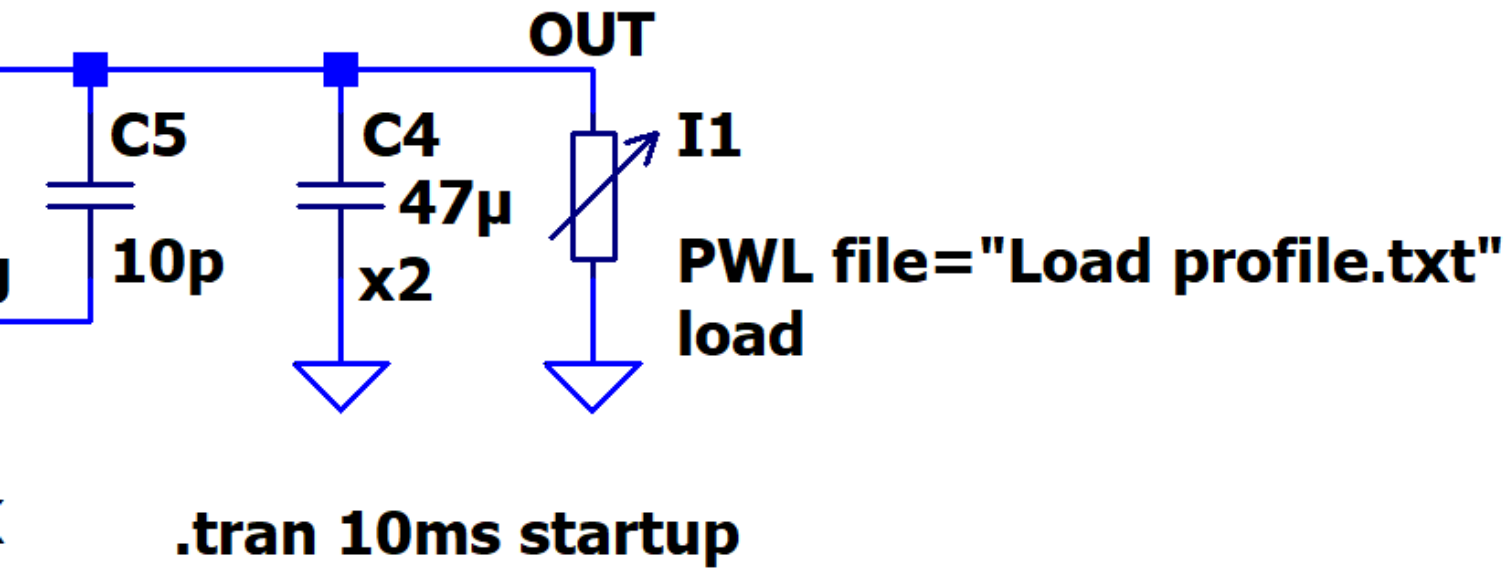


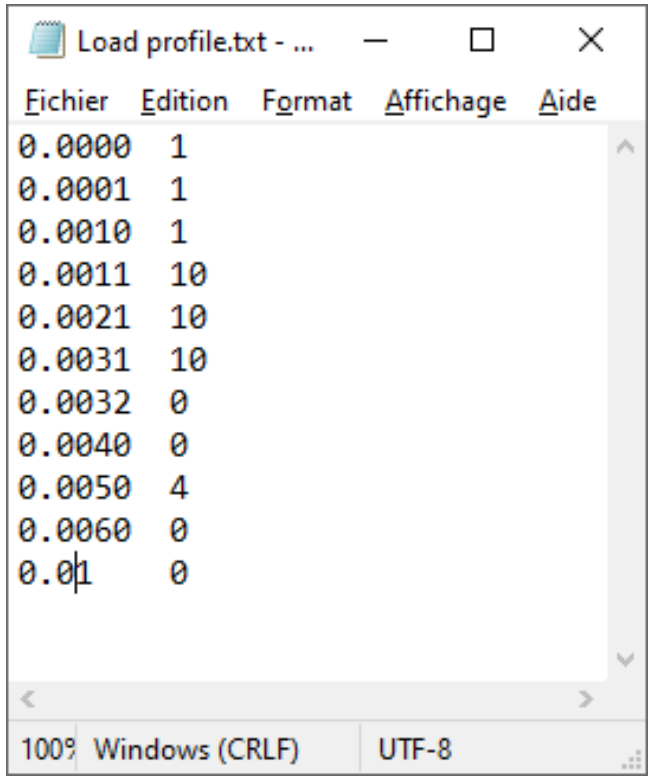


USAGE OF PWL TEST VECTORS

Generation, acquisition, and use of PWL vectors

- Converters are stressed by variations of the output load
- Load transient simulation cuts down validation time

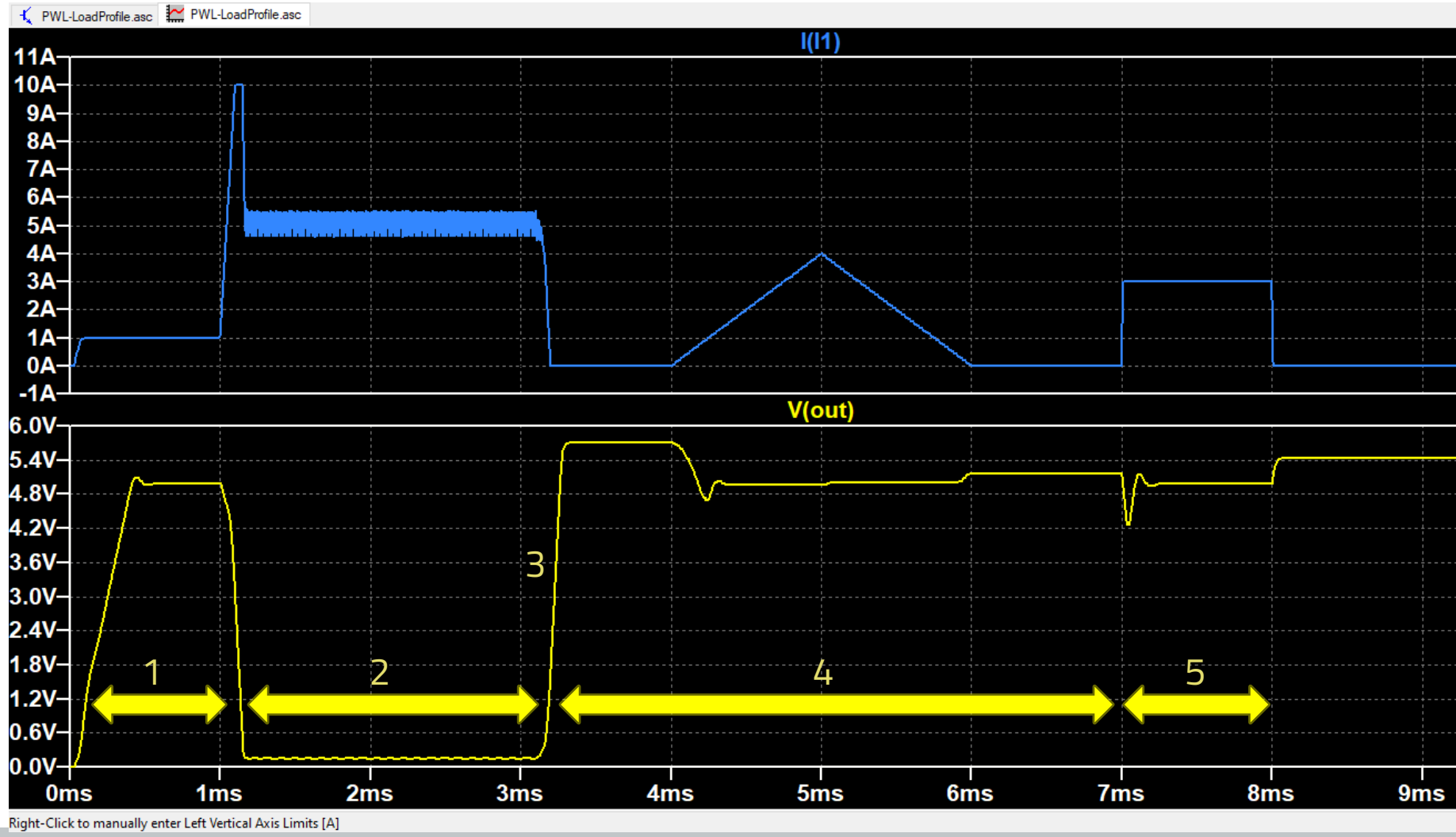




USAGE OF PWL TEST VECTORS

Generation, acquisition, and use of PWL vectors

- Only a run of simulation to validate multiple scenarios
  - Startup (1)
  - Overload (2)
  - Short circuit recovery (3)
  - Light load & No load (4)
  - Load transients (5)
  - Load Bursts

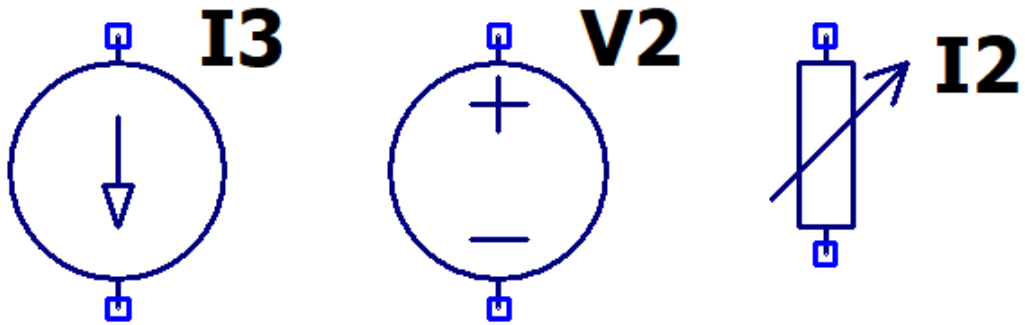


USAGE OF PWL TEST VECTORS

Generation, acquisition, and use of PWL vectors

- Repeat a test vector

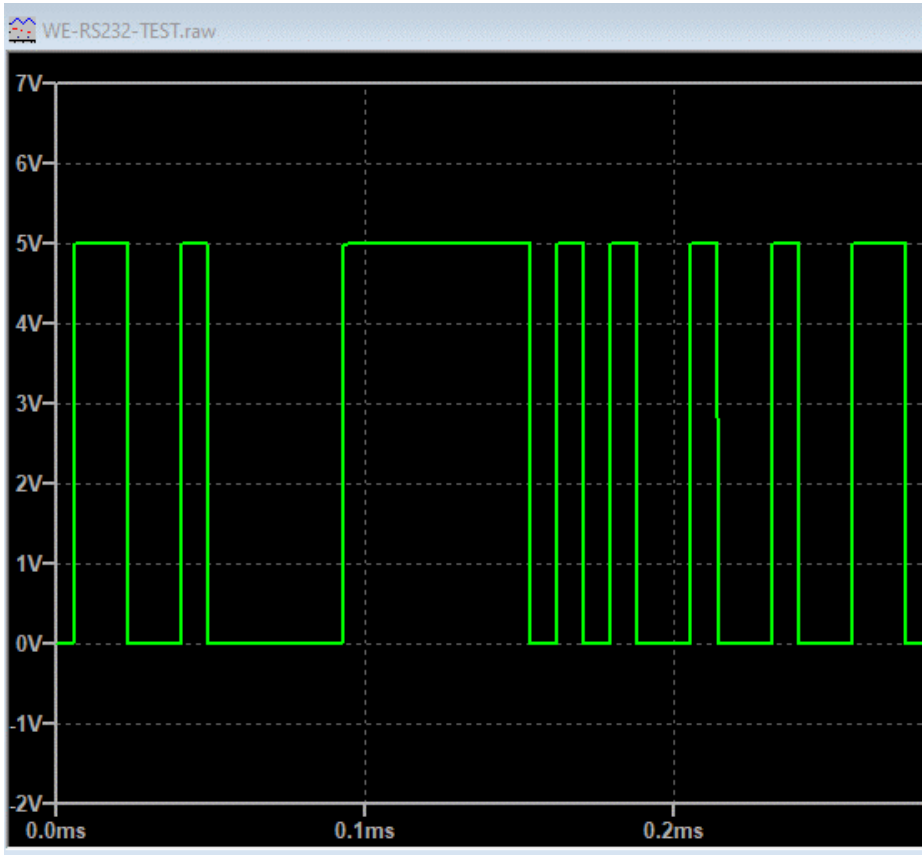
- Applies to many components both sources and sink behavior



USAGE OF PWL TEST VECTORS

Generation, acquisition, and use of PWL vectors

```
// initialize
fprintf (pFile,"%dn\t0\n",0,initialVoltage);
fprintf (pFile,"%dn\t0\n",DELAY,initialVoltage);

// loop for every sample
for (sample = 0 ; sample < NUMBER_OF_SAMPLES ; sample++)
{
   // Compute a sample
   logicLevel = rand() % 2;

   // print sample
   if (logicLevel == 0)
   {
            fprintf (pFile,"%fn\t%5.2f\n",(sample*symbolTime)+DELAY+transitionTime,voltageLow);
            fprintf (pFile,"%fn\t%5.2f\n",((sample+1)*symbolTime)+DELAY-transitionTime,voltageLow);
            printf ("%fn\t%5.2f\n",(sample*symbolTime)+DELAY,voltageLow);
   }
   else
   {
            fprintf (pFile,"%fn\t%5.2f\n",(sample*symbolTime)+DELAY+transitionTime,voltageHigh);
            fprintf (pFile,"%fn\t%5.2f\n",((sample+1)*symbolTime)+DELAY-transitionTime,voltageHigh);
            printf ("%fn\t%5.2f\n",(sample*symbolTime)+DELAY,voltageHigh);
   }
}
```

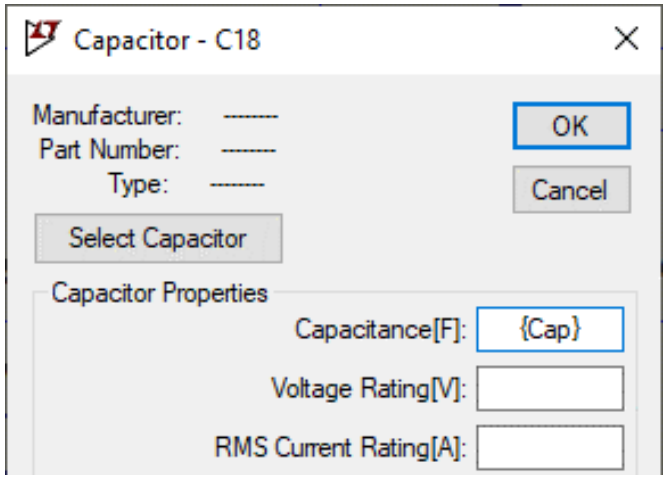


ITERATION AND FUNCTIONAL VALIDATION

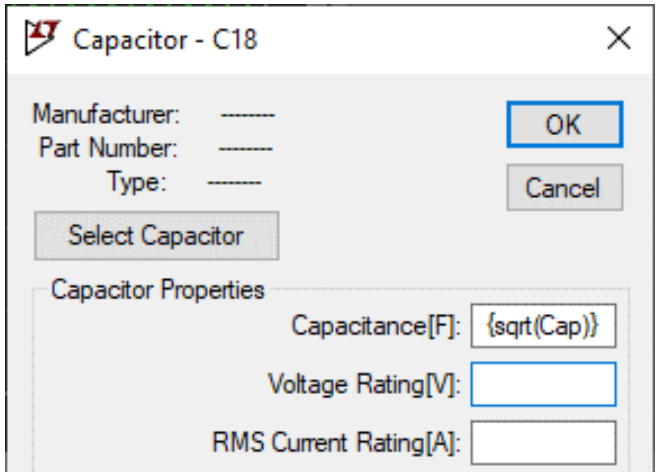
Step and parameters

.param Cap=2.2nF On Sheet Statement using the SPICE directive command

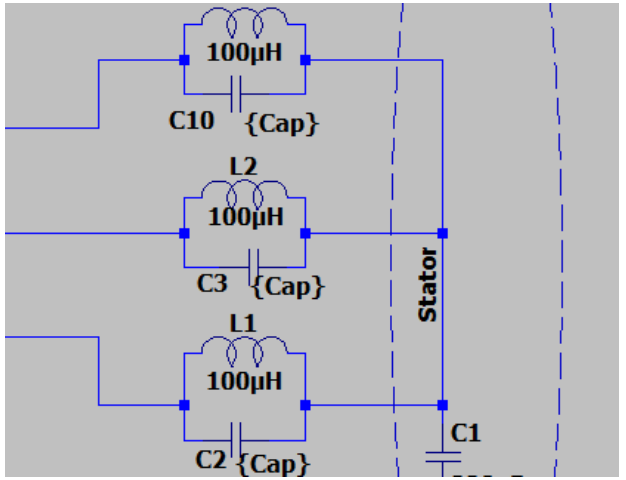
Usage in a component



Usage in an expression



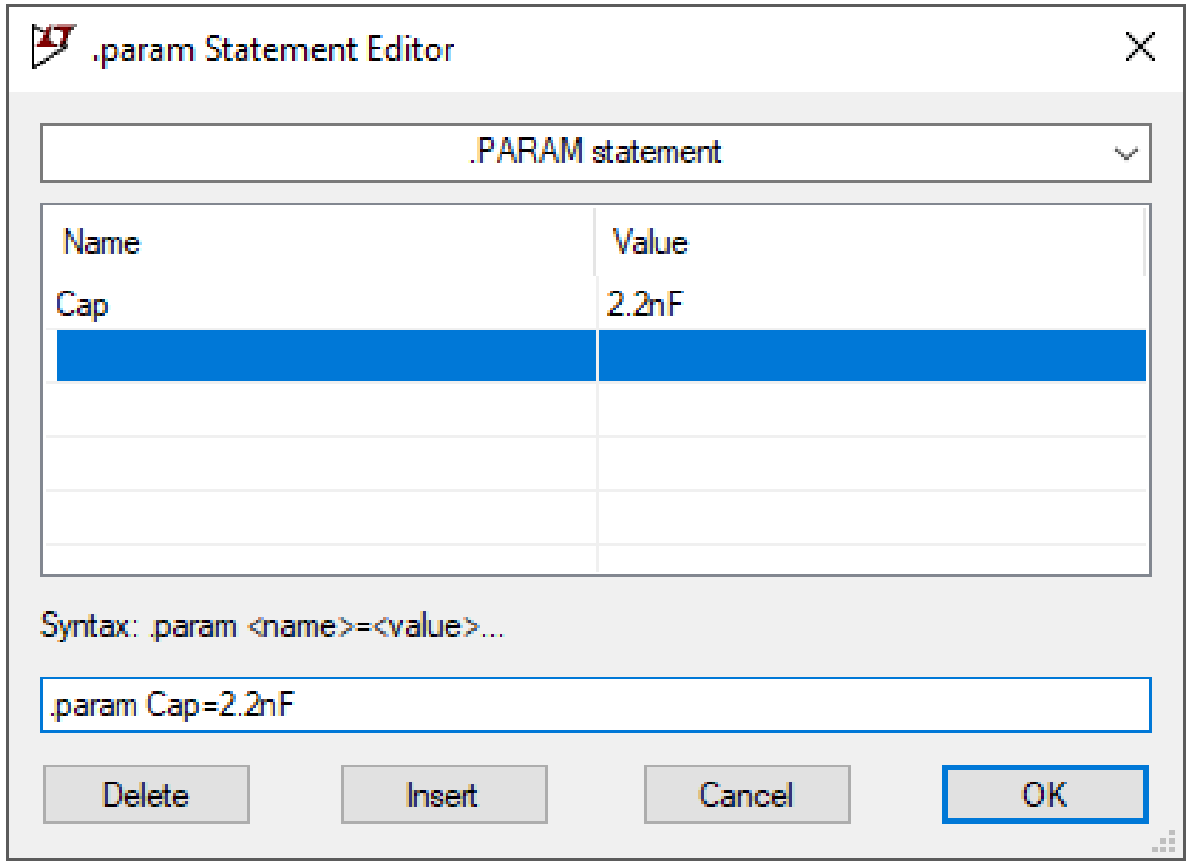
Usage in multiple components

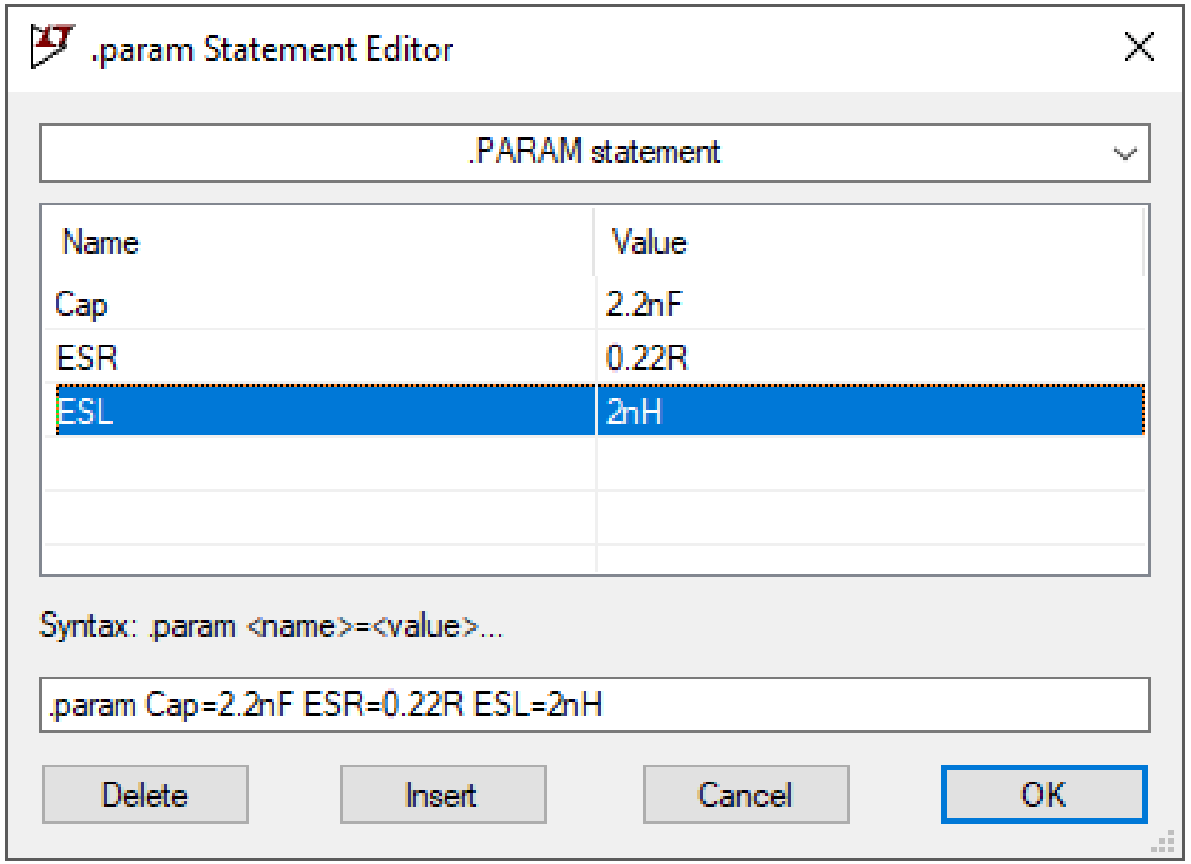


ITERATION AND FUNCTIONAL VALIDATION

Step and parameters

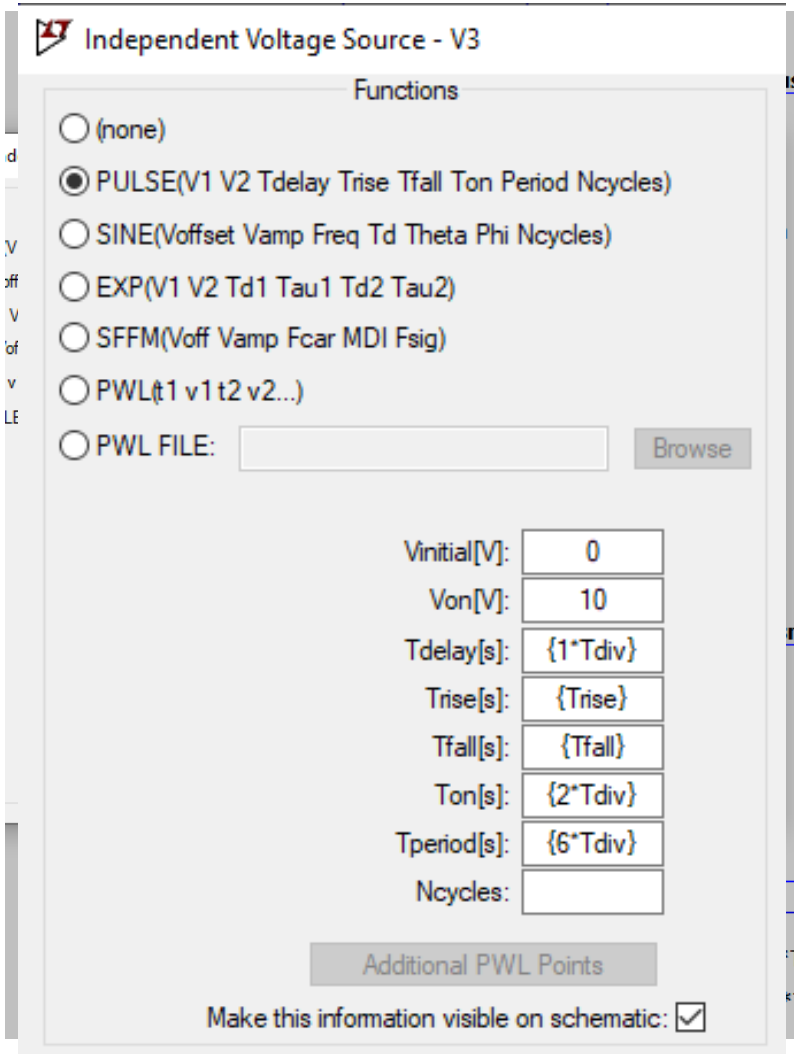
Hint : click *.param* statement to have access to an Editor

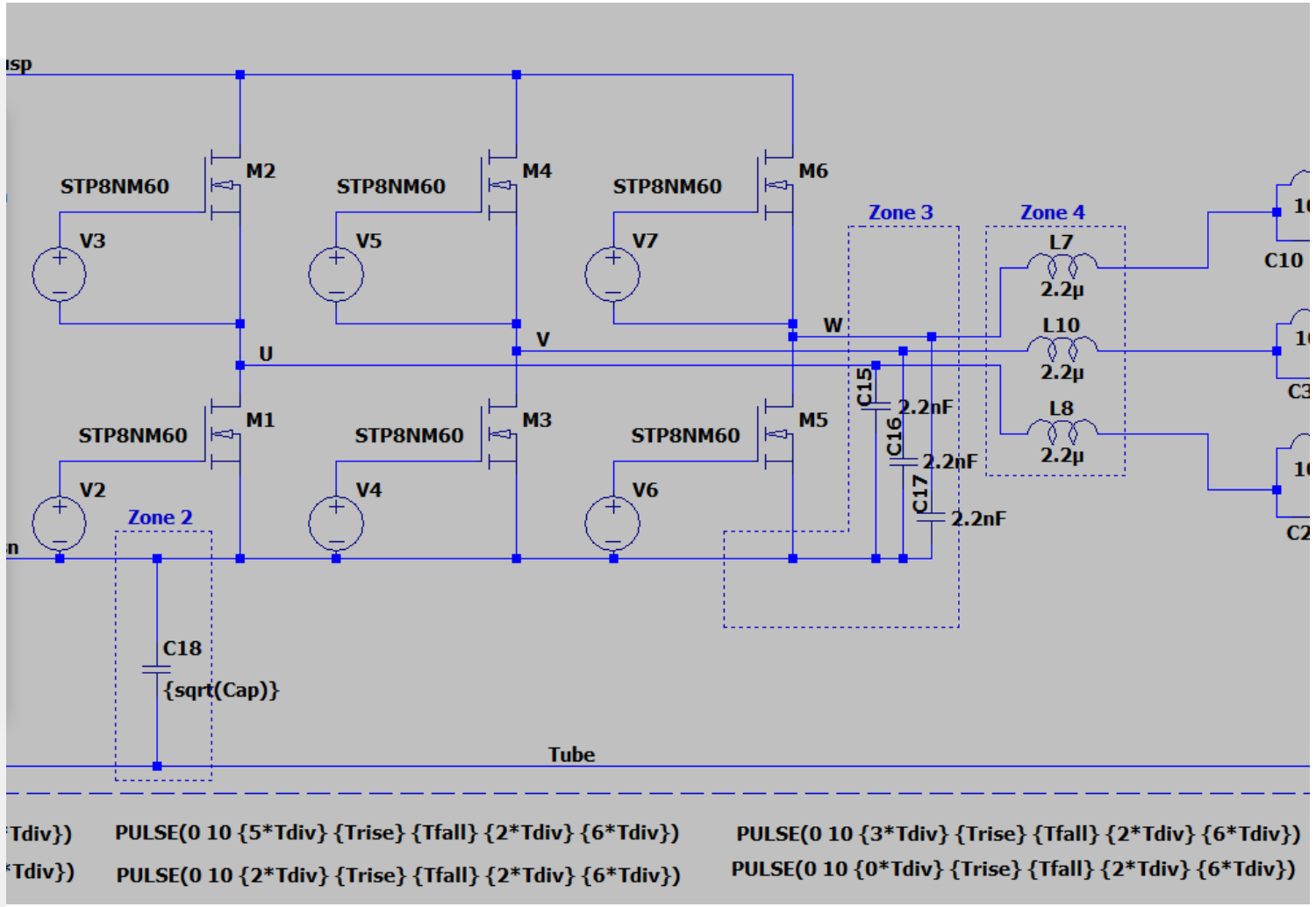




ITERATION AND FUNCTIONAL VALIDATION

Step and parameters

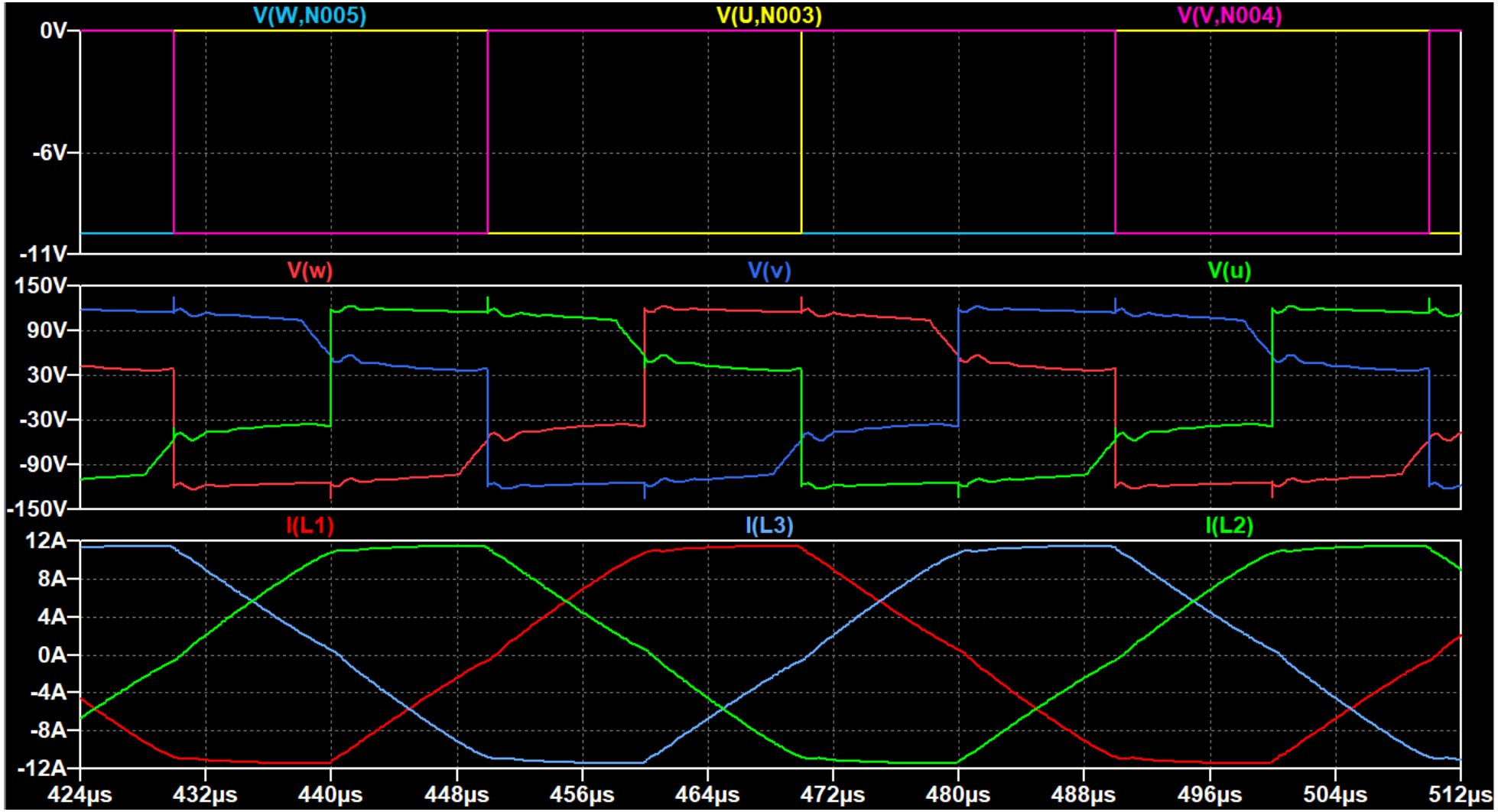




ITERATION AND FUNCTIONAL VALIDATION

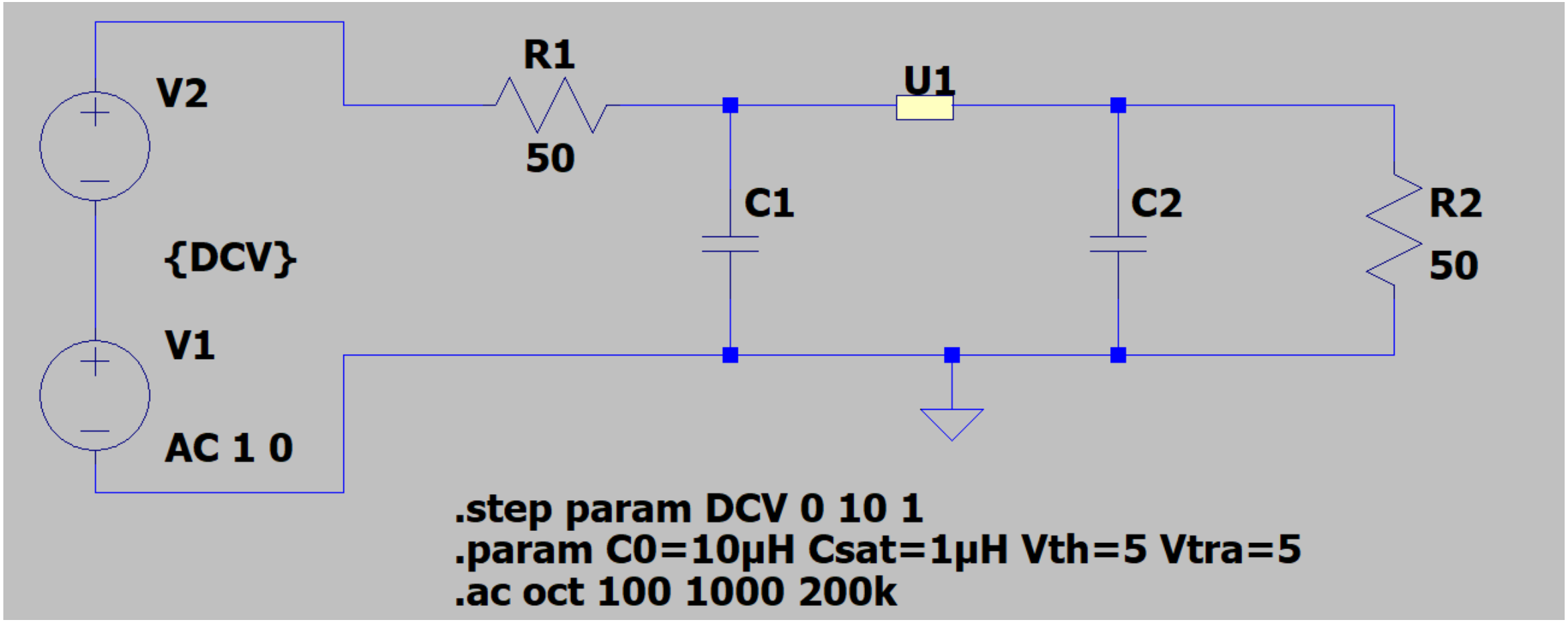
Step and parameters

- You can easily adjust :
  - Slope
  - Deadtime
  - Frequency



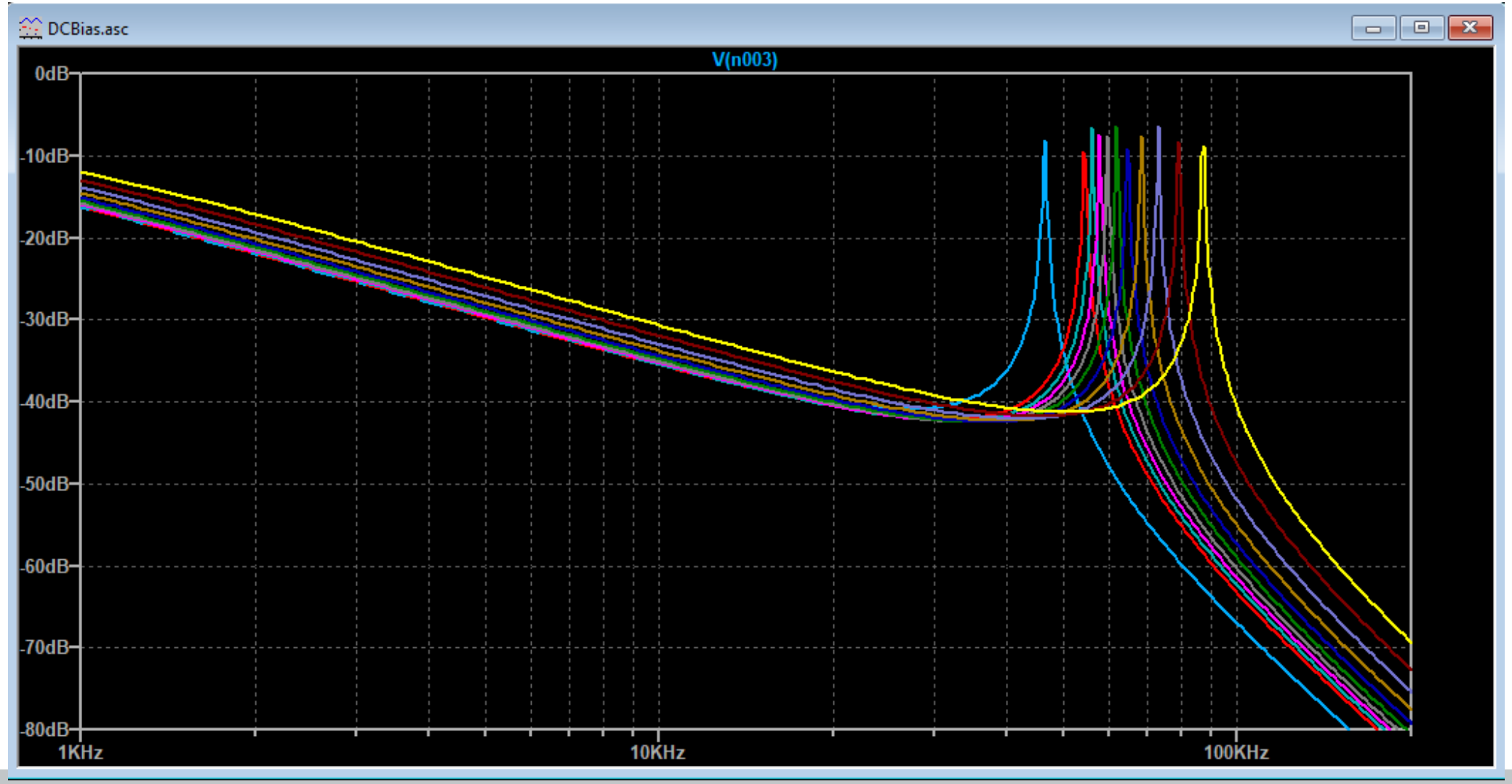
ITERATION AND FUNCTIONAL VALIDATION

Step and parameters



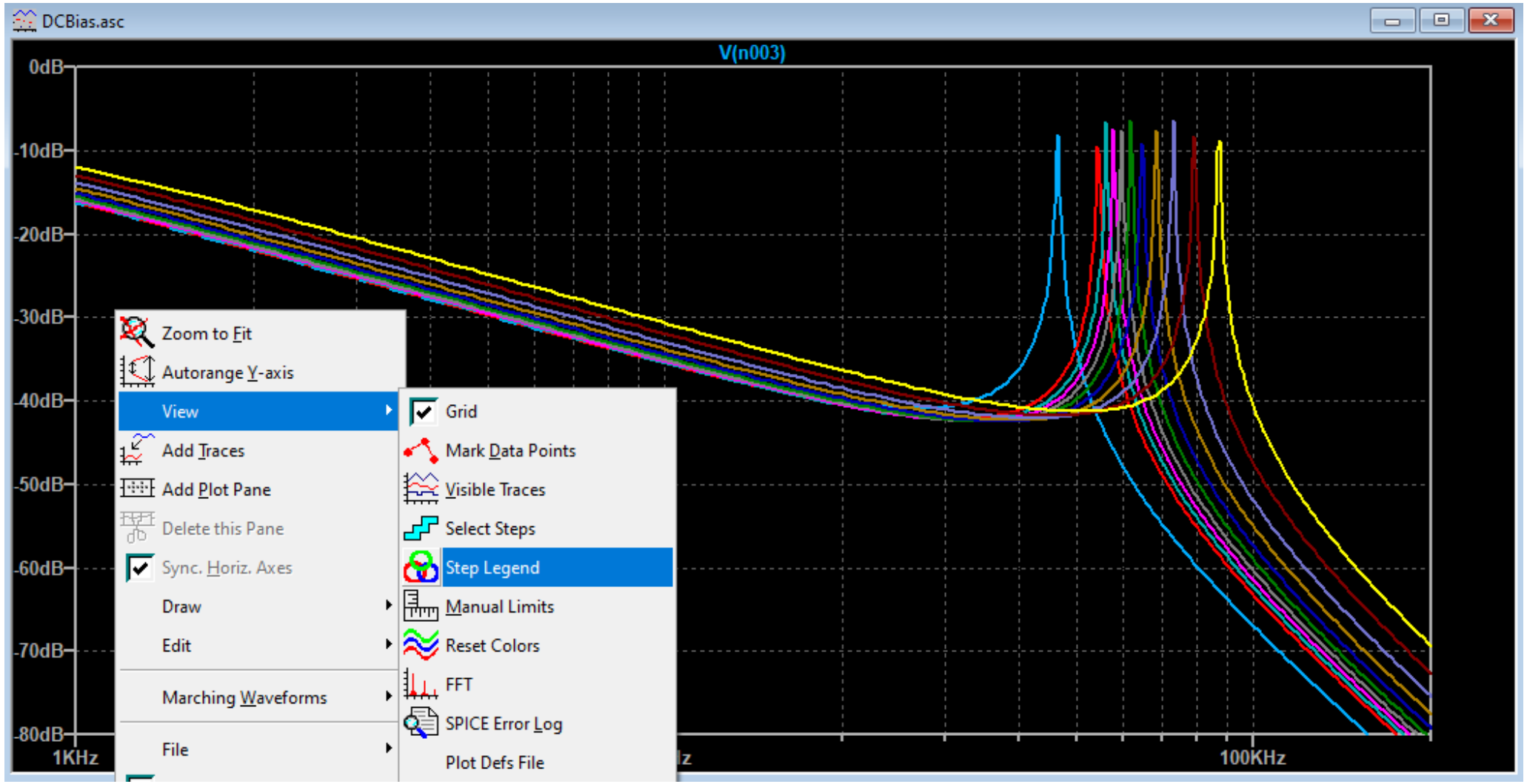
ITERATION AND FUNCTIONAL VALIDATION

Step and parameters



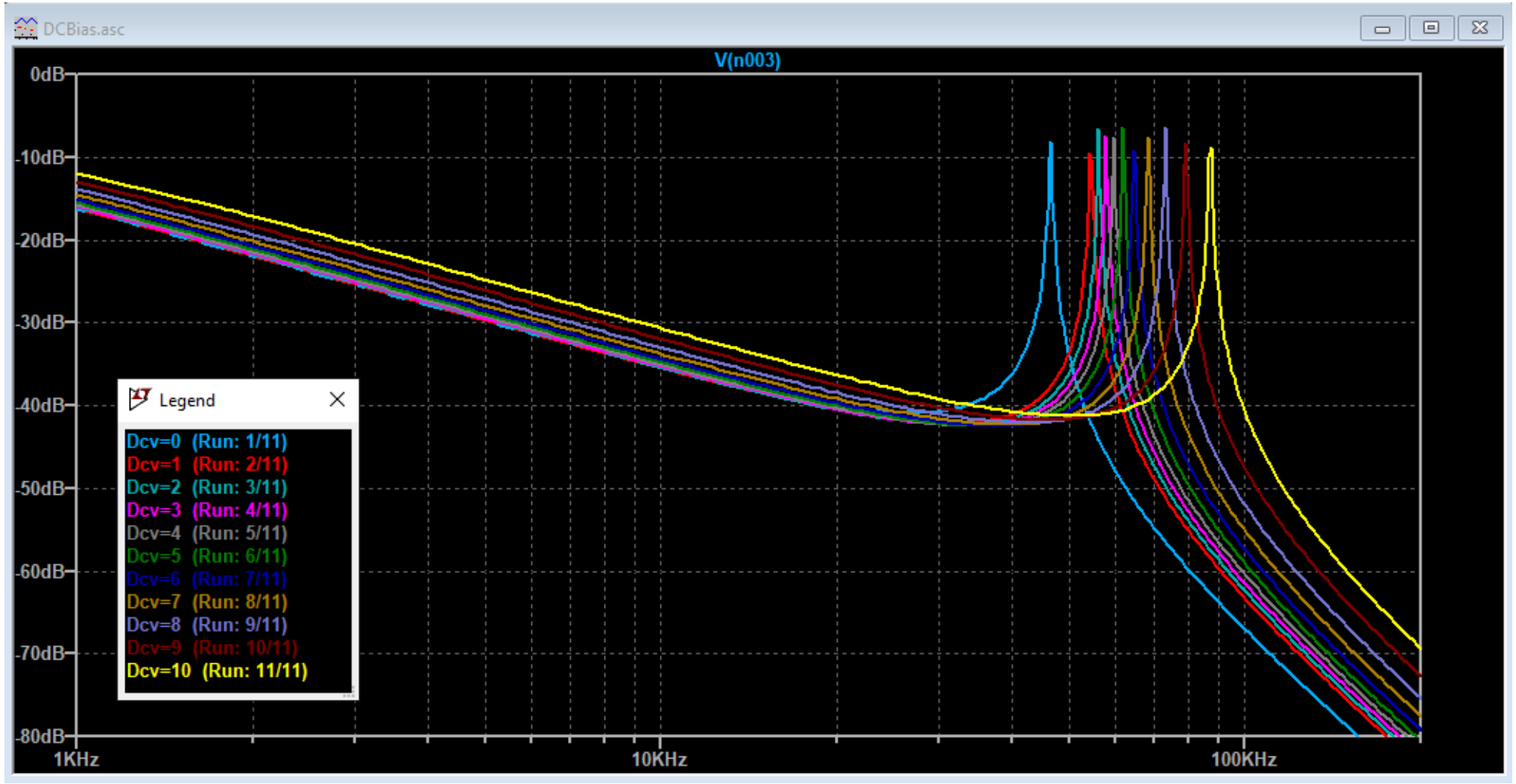
ITERATION AND FUNCTIONAL VALIDATION

Step and parameters



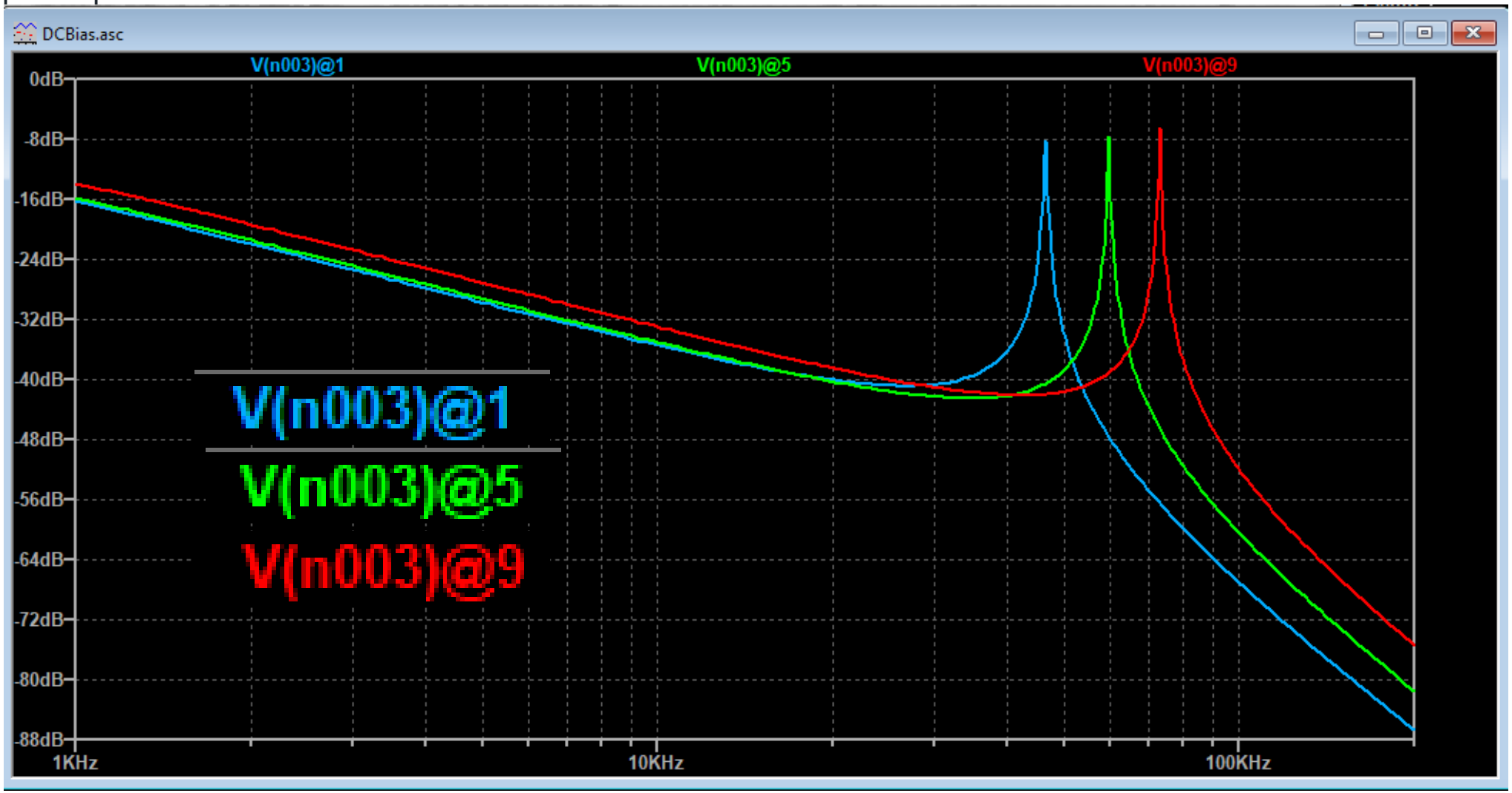
ITERATION AND FUNCTIONAL VALIDATION

Step and parameters



ITERATION AND FUNCTIONAL VALIDATION

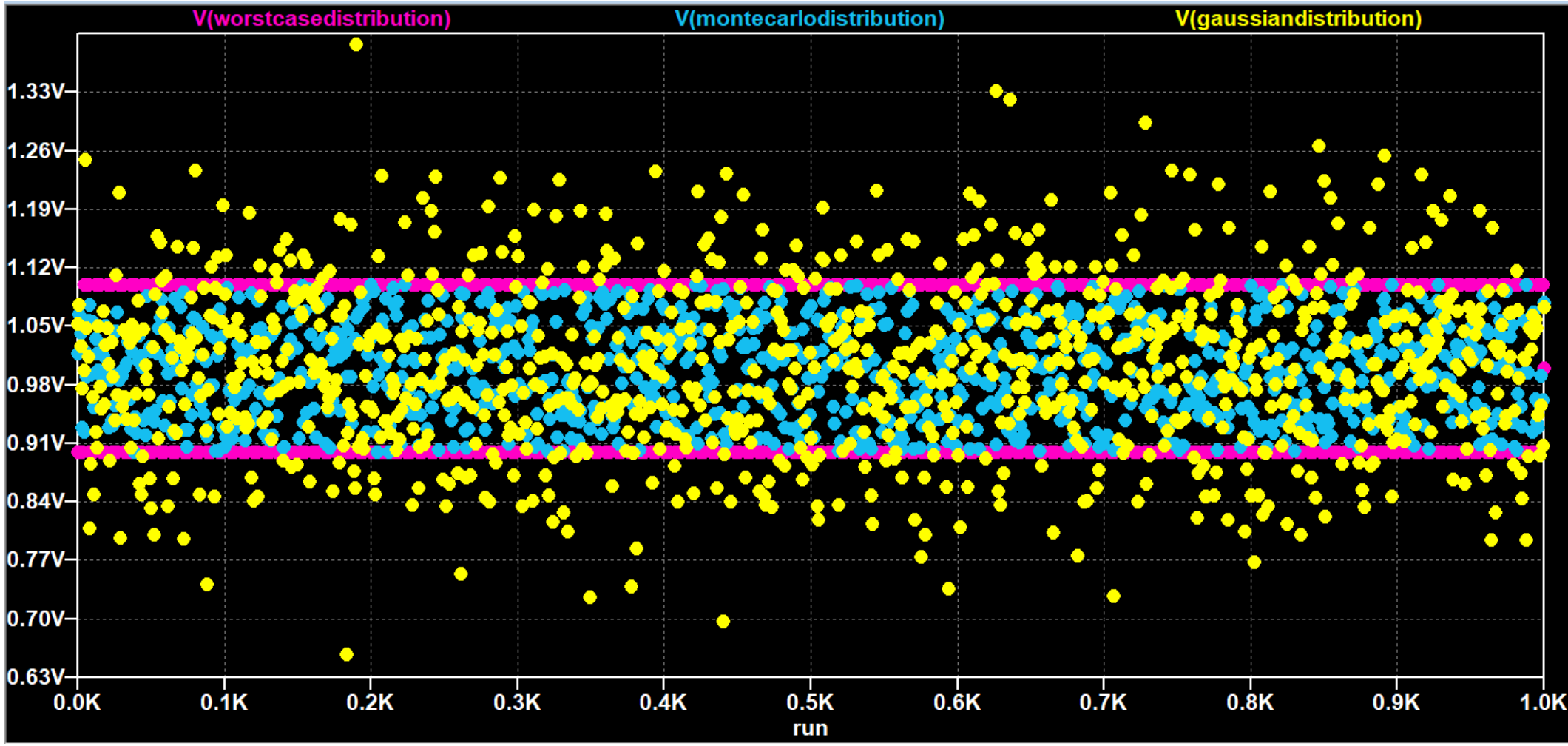
Step and parameters



MONTE CARLO, WORST CASE AND GAUSSIAN DISTRIBUTION

Adding some randomization to the simulation iterations

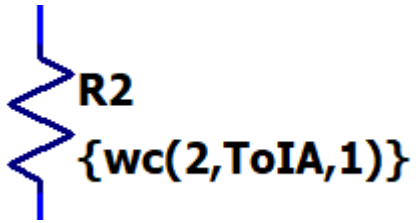
- Useful for statistical validation over tolerance range
- Ideally suited to run during
  - Coffee breaks
  - Lunch breaks
  - Nights
- Ltspice enables plotting any variable versus simulation run number

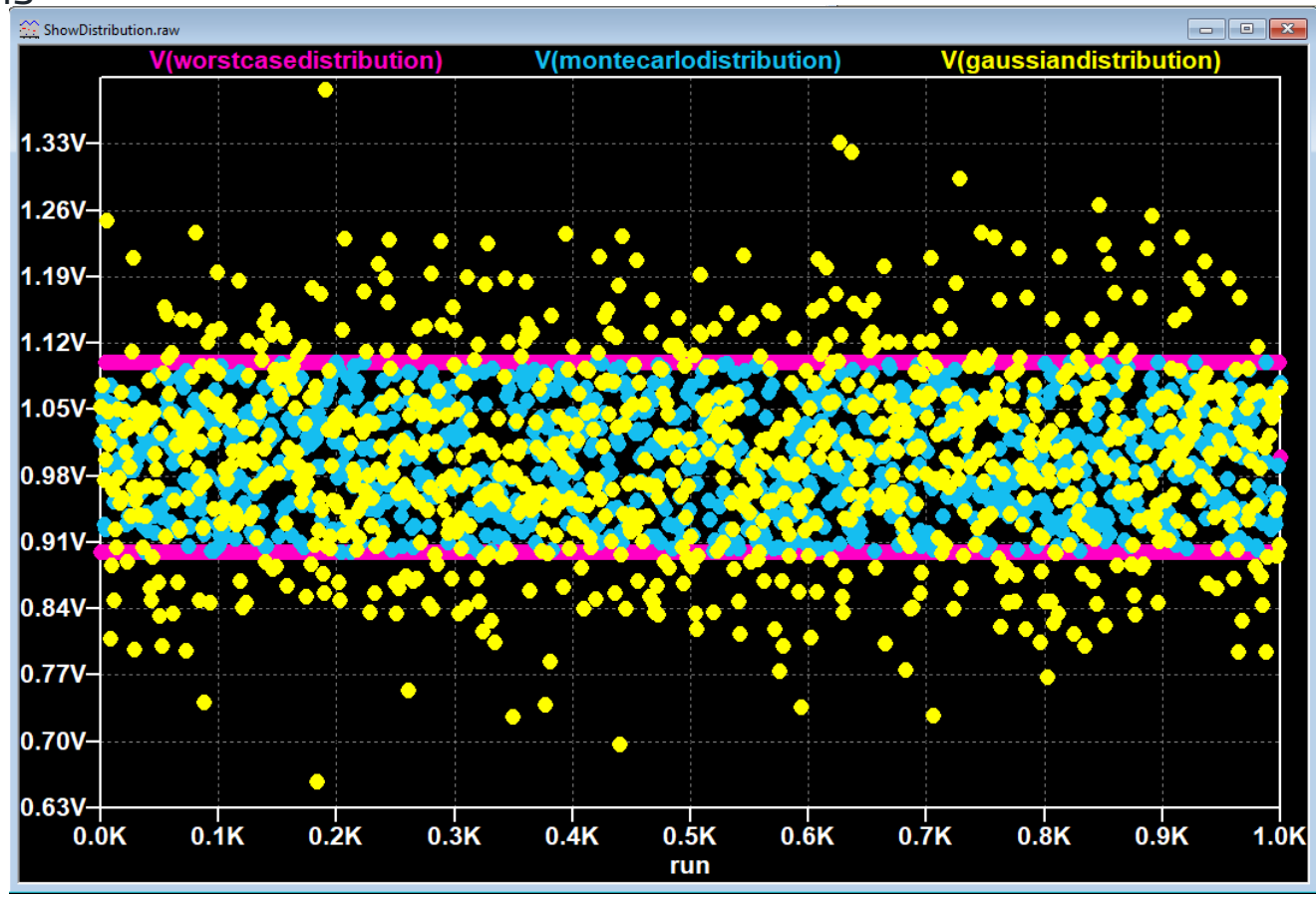


MONTE CARLO, WORST CASE AND GAUSSIAN DISTRIBUTION

Adding some randomization to the simulation iterations

- **Worst case distribution** is used for quick validation at :
  - nominal value
  - nominal value + tolerance
  - nominal value – tolerance
- It's the fastest way to perform a gross validation
- It can test combinations of components tolerances as long as their index is different
- This function **isn't built in** Ltspice, it requires a copy paste as spice directive (.func)



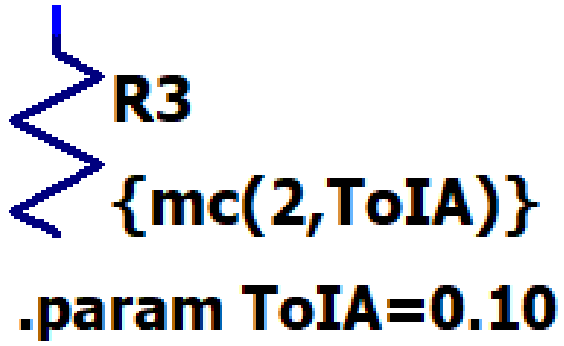


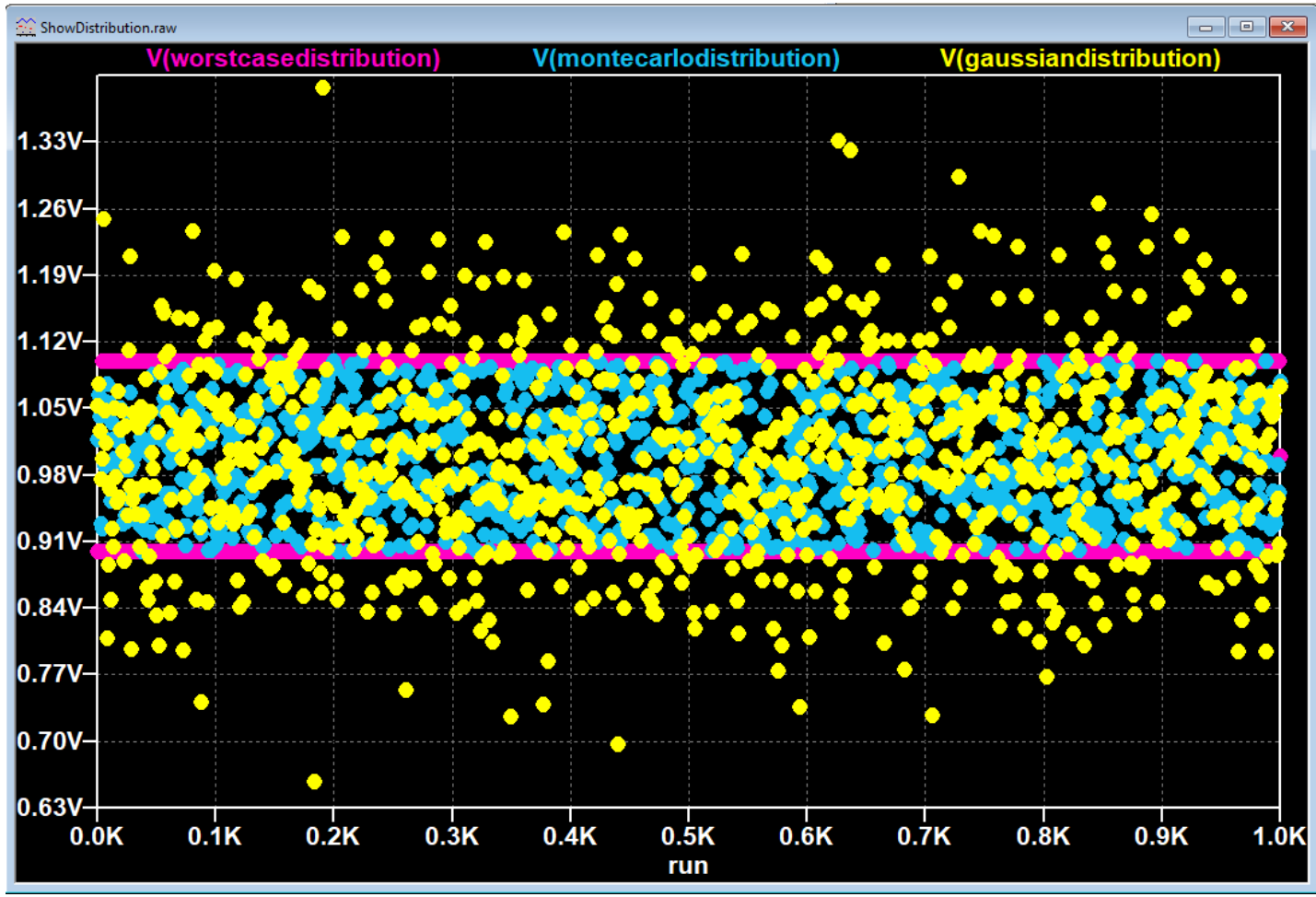

```
.func wc(nom,tol,index) if(run==numruns,nom,if(binary(run,index),nom*(1+tol),nom*(1-tol)))
.func binary(run,index) floor(run/(2**index))-2*floor(run/(2**(index+1)))
```

MONTE CARLO, WORST CASE AND GAUSSIAN DISTRIBUTION

Adding some randomization to the simulation iterations

- **Monte Carlo distribution** is used for validation within a tolerance range
- It's the most convenient way to perform validation of components having a **known tolerance.**
- **It uses a Uniform distribution of samples within tolerance**
- Many simulation runs are required for a good coverage
- This function **is built in** Ltspice

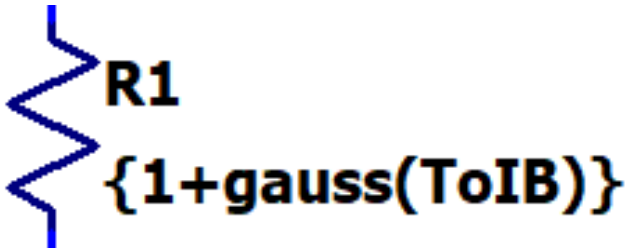




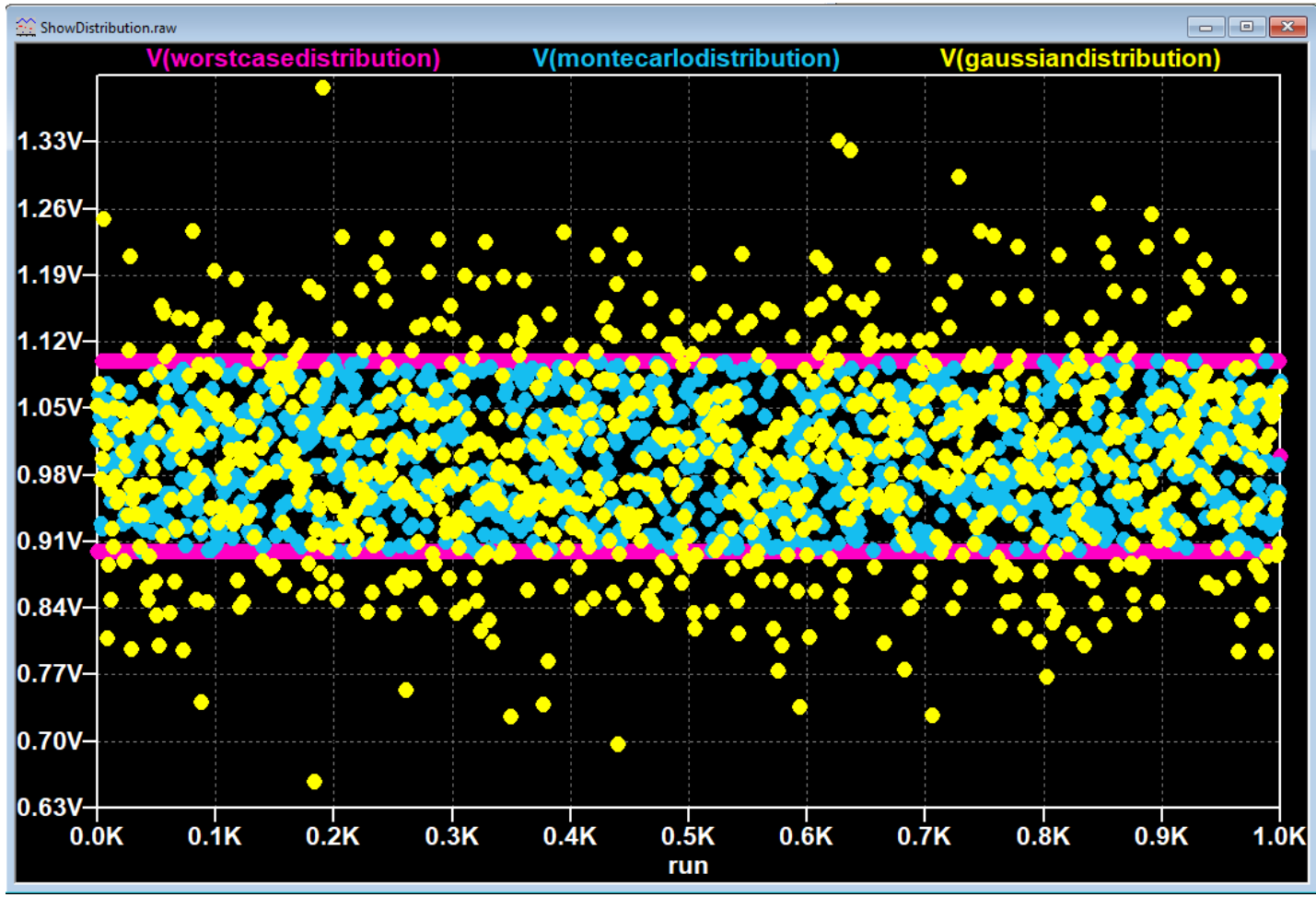
MONTE CARLO, WORST CASE AND GAUSSIAN DISTRIBUTION

Adding some randomization to the simulation iterations

- **Gaussian distribution** is used for validation of components having **bell shaped distribution**
- It's the most convenient way to perform validation of unsorted components
- Many simulation runs are required for a good coverage
- This function **is built in** Ltspice

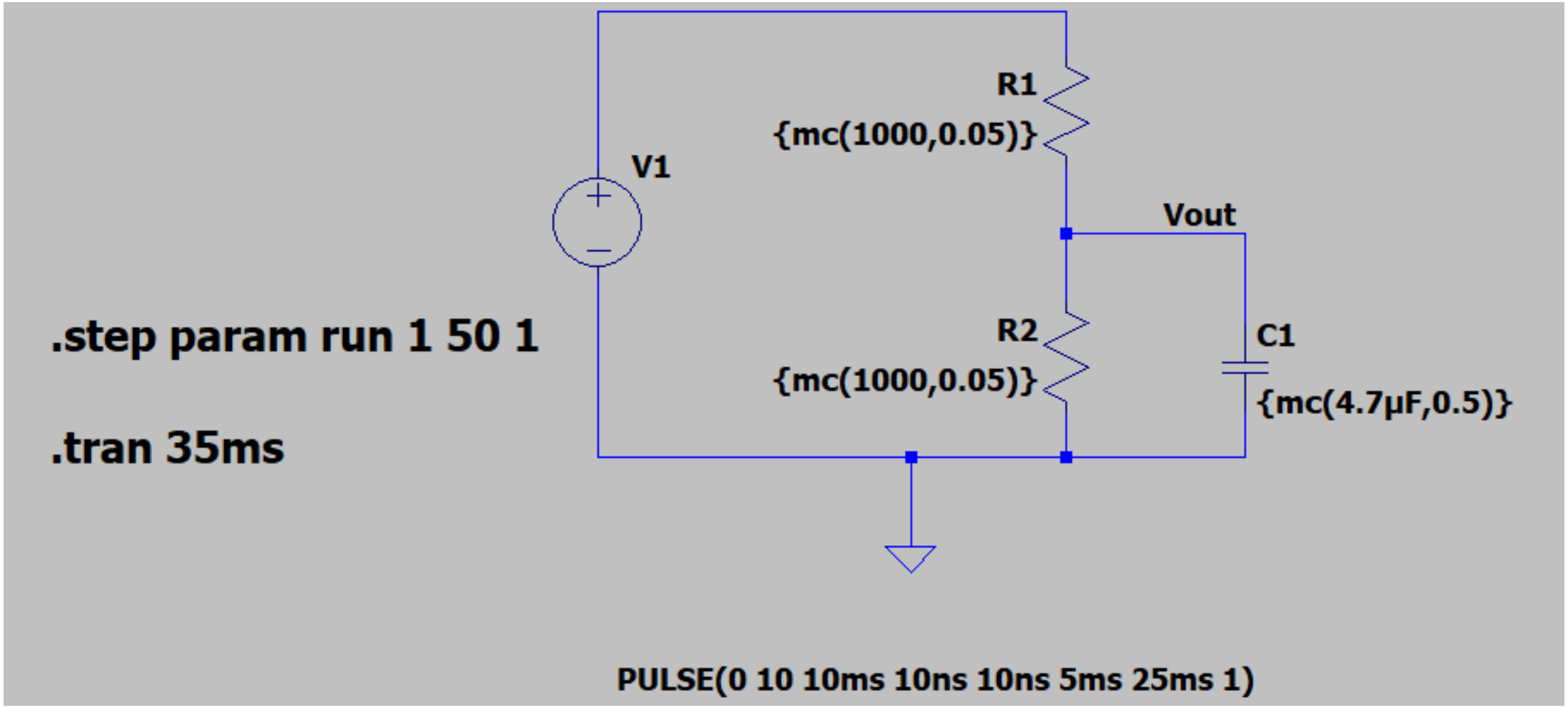



```
.param ToIB=0.10
```



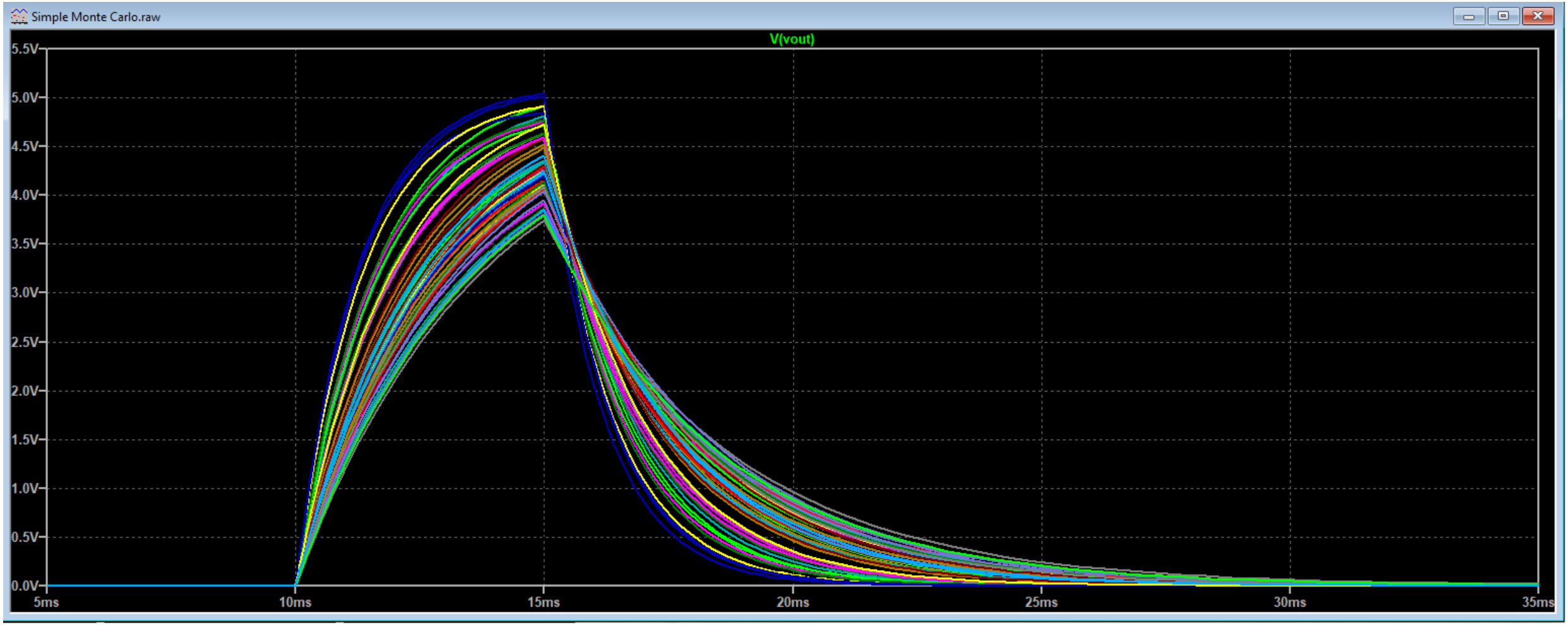
MONTE CARLO, WORST CASE AND GAUSSIAN DISTRIBUTION

Adding some randomization to the simulation iterations



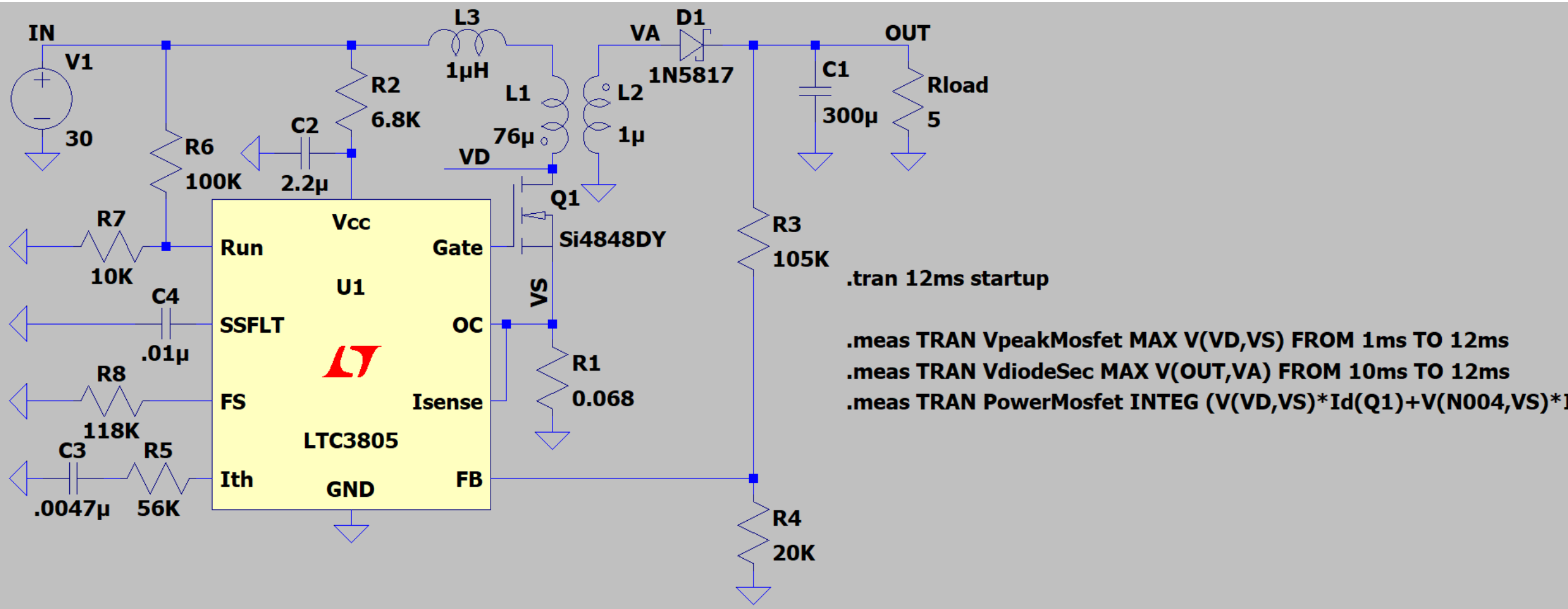
MONTE CARLO, WORST CASE AND GAUSSIAN DISTRIBUTION

Adding some randomization to the simulation iterations



ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement



ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement

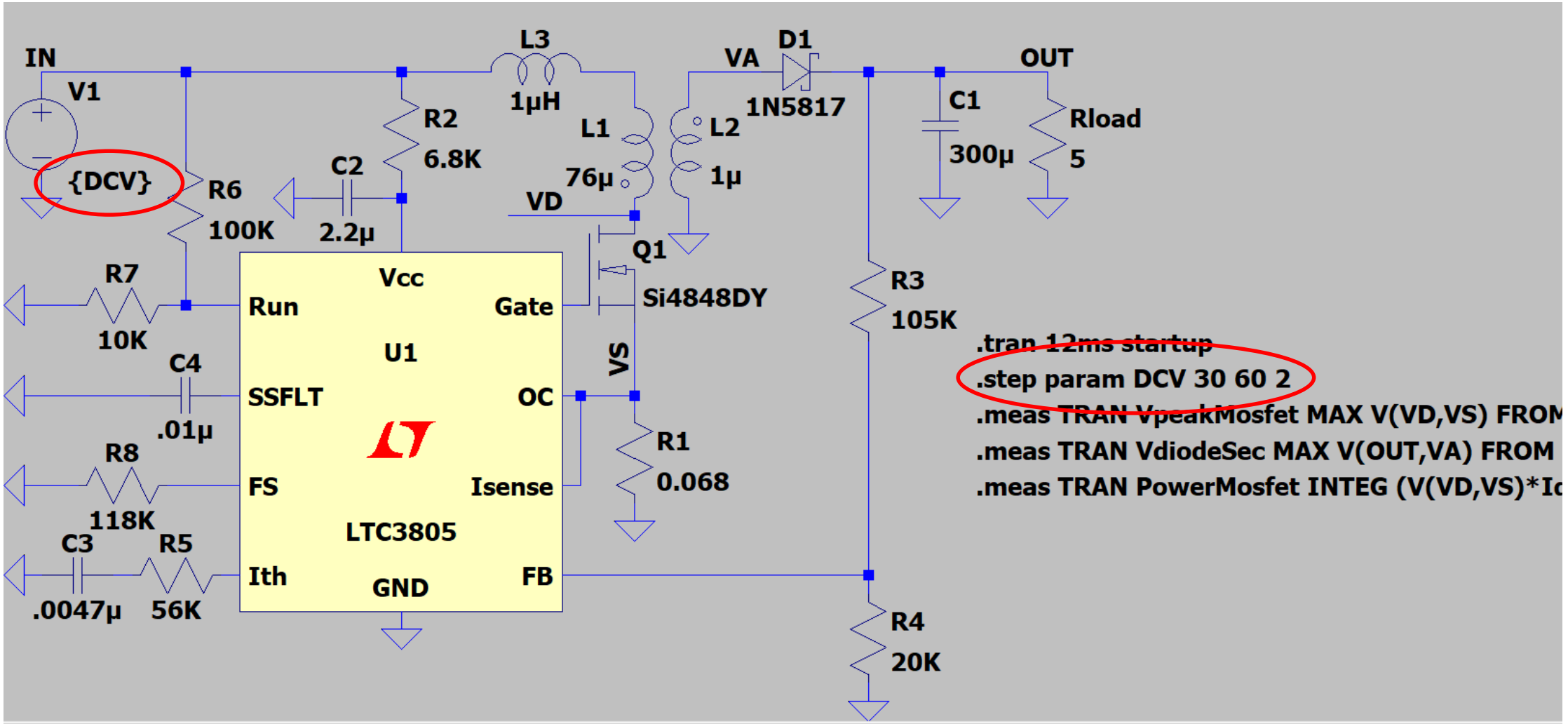
```
.meas TRAN VpeakMosfet MAX V(VD,VS) FROM 1ms TO 12ms
.meas TRAN VdiodeSec MAX V(OUT,VA) FROM 1ms TO 12ms
.meas TRAN PowerMosfet INTEG (V(VD,VS)*Id(Q1)+V(N004,VS)*Ig(Q1))*1000 FROM 11ms TO 12ms
```

SPICE Error Log: C:\Users\sylvain.lebras\Documents\LTspiceXVII\examples\jigs\3805.log

```
Circuit: * C:\Users\sylvain.lebras\Documents\LTspiceXVII\examples\jigs\3805.asc

Direct Newton iteration for .op point succeeded.
Ignoring empty pin current: Ix(u1:6)
Ignoring empty pin current: Ix(u1:6)

vdiodesec: MAX(v(out,va))=8.64541 FROM 0.01 TO 0.012
vpeakmosfet: MAX(v(vd,vs))=125.936 FROM 0.001 TO 0.012
powermosfet: INTEG((v(vd,vs)*id(q1)+v(n004,vs)*ig(q1))*1000)=0.541502 FROM 0.011 TO 0.012
```

ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement

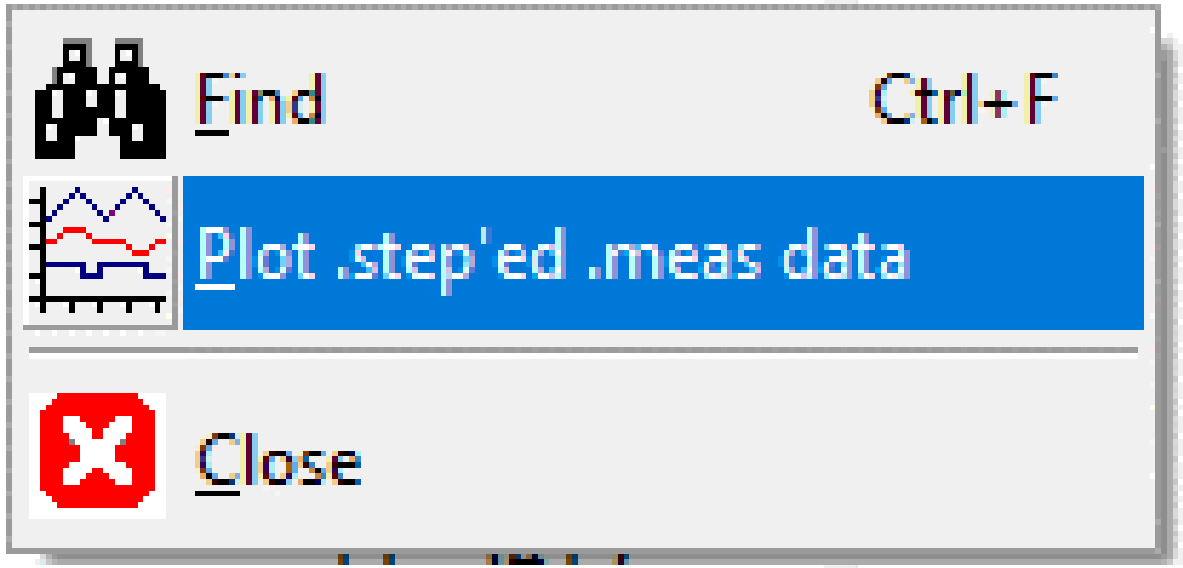


ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement

SPICE Error Log: C:\Users\sylvain.lebras\Documents\LTspiceXVII\examples\jigs\3805.log

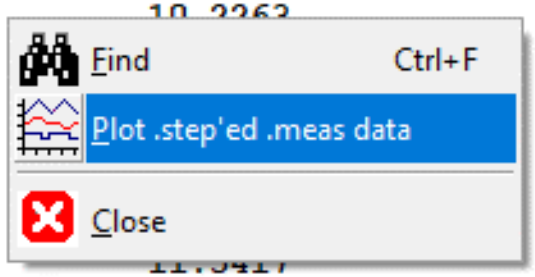
```
Measurement: vdiodesec
  step             MAX(v(out,va))      FROM              TO
     1             8.88332             0.001             0.012
     2             9.3101              0.001             0.012
     3             9.40636             0.001             0.012
     4             9.81061             0.001             0.012
     5             9.9069              0.001             0.012
     6             10.2263             0.001             0.012
     7             10.4535             0.001             0.012
     8             10.6544             0.001             0.012
     9             10.9606             0.001             0.012
    10             11.0616             0.001             0.012
    11             11.3662             0.001             0.012
    12             11.5417             0.001             0.012
    13             11.8517             0.001             0.012
    14             12.0913             0.001             0.012
    15             12.3436             0.001             0.012
    16             12.5908             0.001             0.012

Measurement: vpeakmosfet
  step             MAX(v(vd,vs))       FROM              TO
     1             125.936             0.001             0.012
     2             128.186             0.001             0.012
     3             129.469             0.001             0.012
     4             131.572             0.001             0.012
     5             133.696             0.001             0.012
     6             135.551             0.001             0.012
```

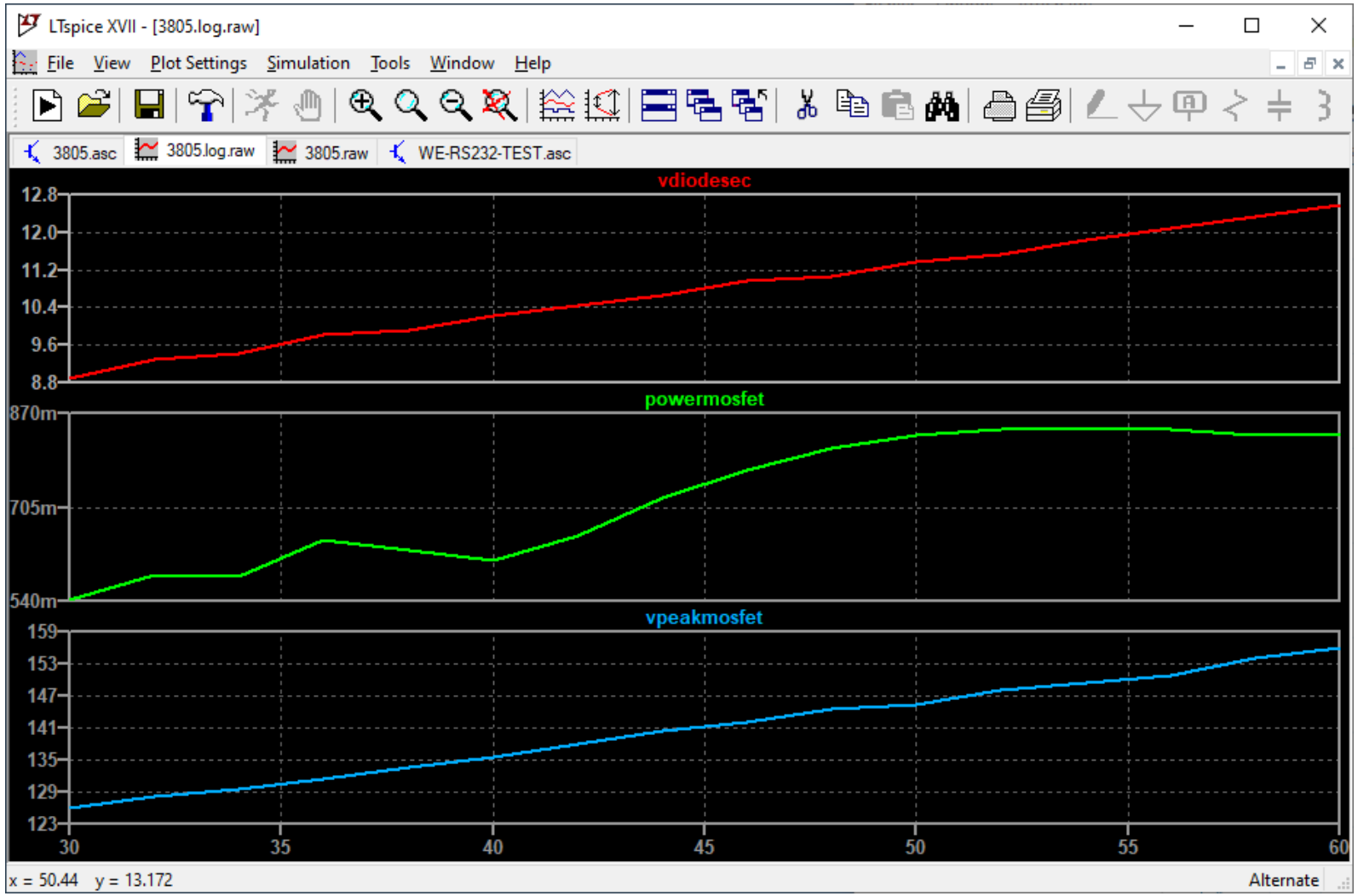


ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement

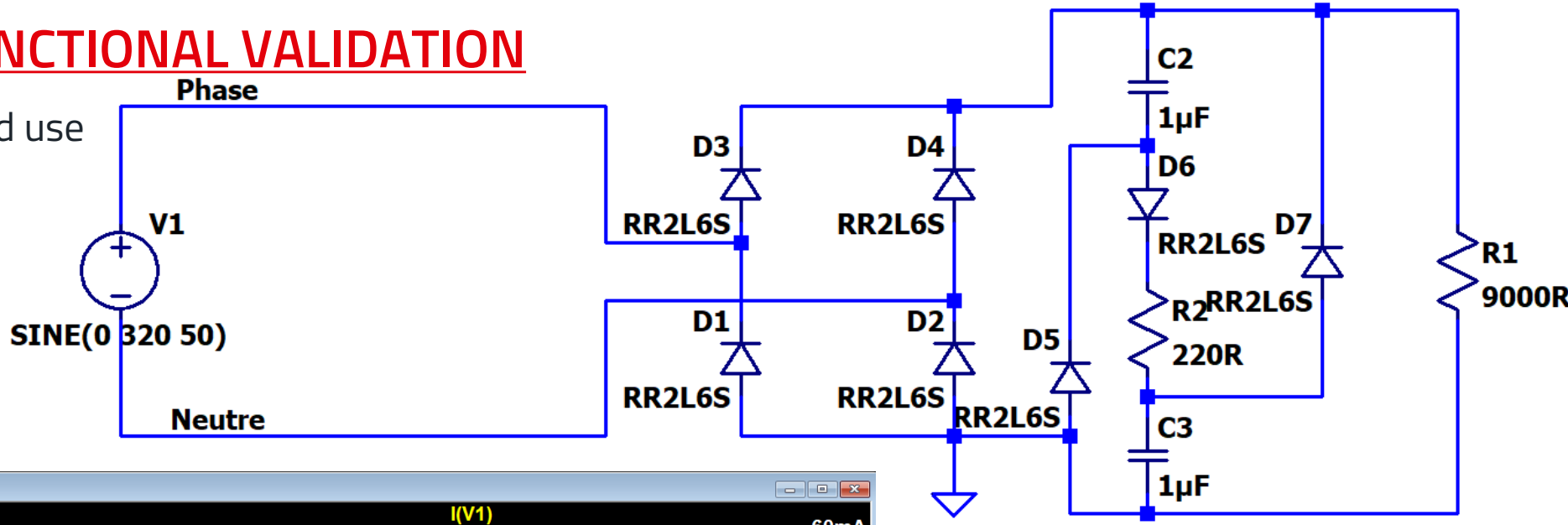
```
Measurement: vdiodesec
   step            MAX(v(out,va))        FROM
      1            8.88332               0.001
      2            9.3101                0.001
      3            9.40636               0.001
      4            9.81061               0.001
      5            9.9069                0.001
      6            10.2263               0.001
      7                                  0.001
      8                                  0.001
      9                                  0.001
     10                                  0.001
     11                                  0.001
     12                                  0.001
     13            11.8517               0.001
     14            12.0913               0.001
     15            12.3436               0.001
     16            12.5908               0.001

Measurement: vpeakmosfet
   step            MAX(v(vd,vs))         FROM
      1            125.936               0.001
      2            128.186               0.001
      3            129.469               0.001
```





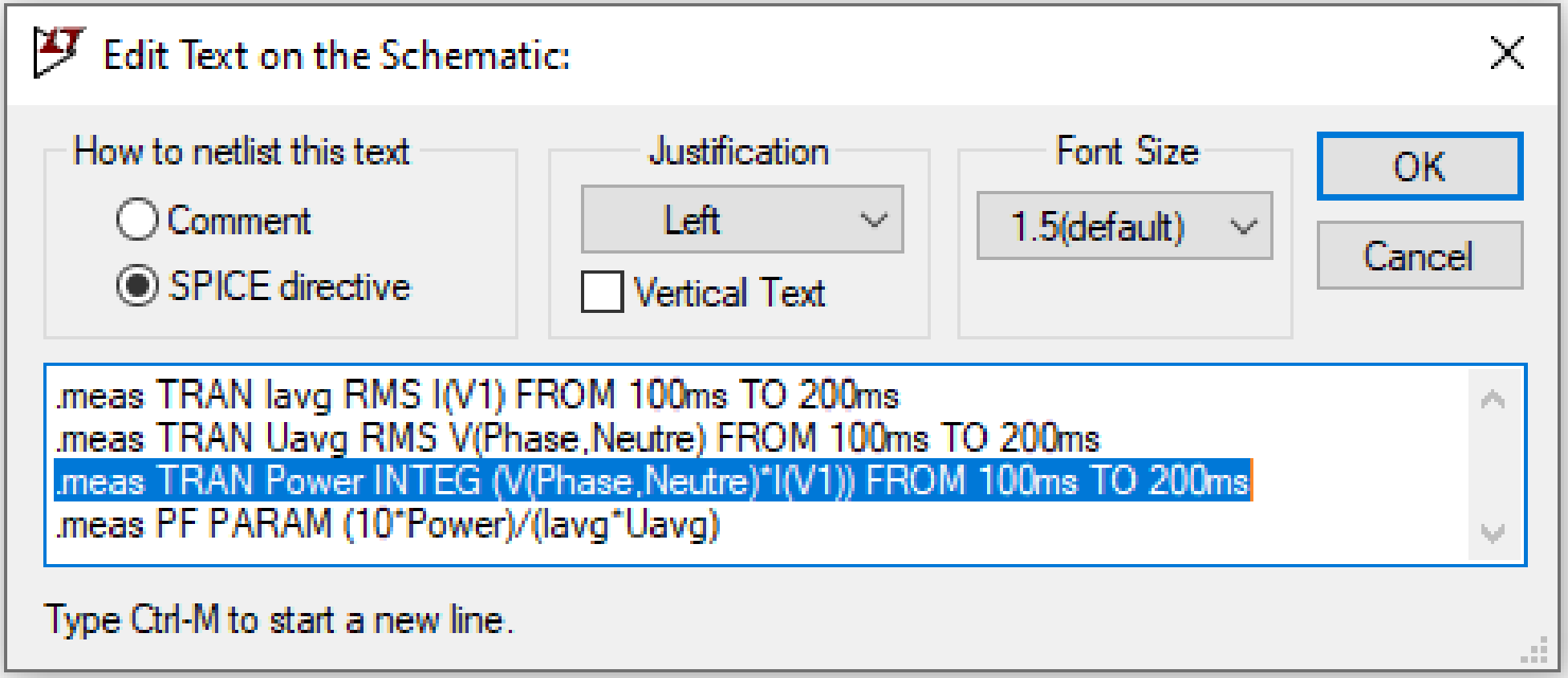
ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement advanced use (power factor)



ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement advanced use (power factor)

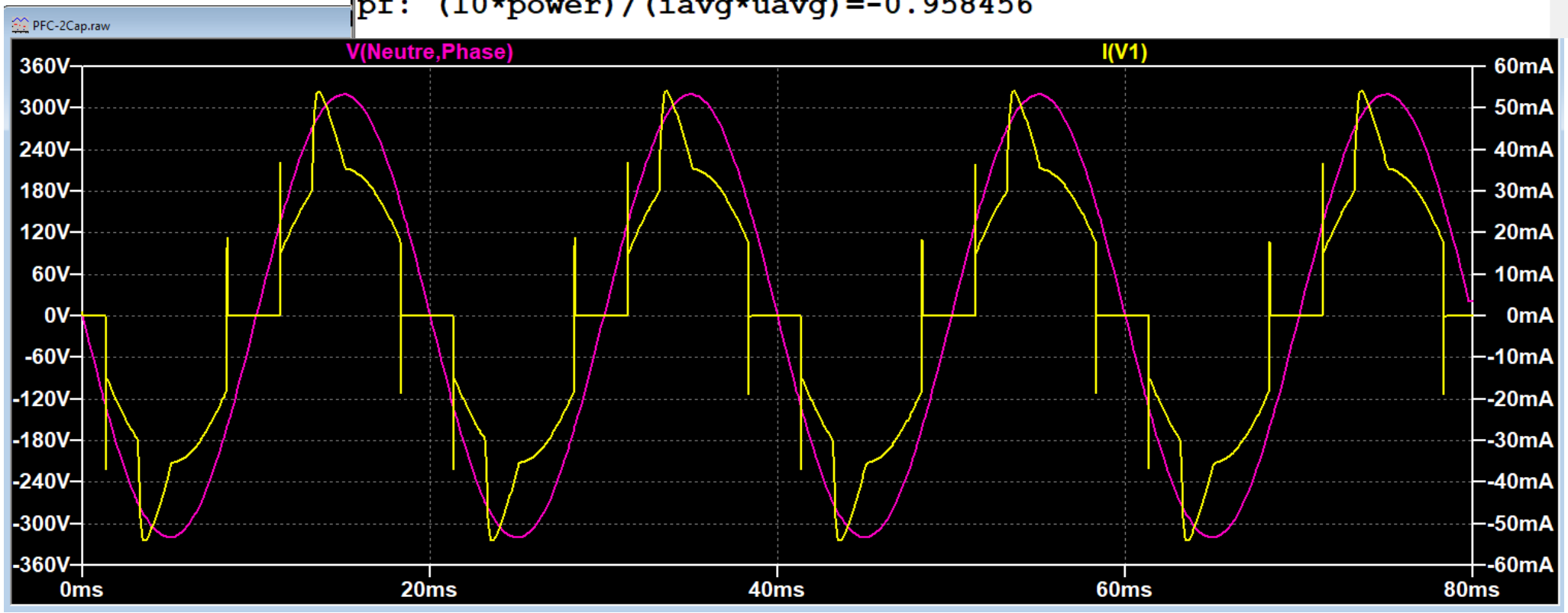


ITERATION AND FUNCTIONAL VALIDATION

.MEAS statement advanced use (power factor)

SPICE Error Log: C:\Users\sylvain.lebras\Documents\Eaton\PFC-2Cap.log

```
iavg: RMS(i(v1))=0.0283024 FROM 0.1 TO 0.2
uavg: RMS(v(phase,neutre))=226.271 FROM 0.1 TO 0.2
power: INTEG(v(phase,neutre)*i(v1))=-0.613797 FROM 0.1 TO 0.2
pf: (10*power)/(iavg*uavg)=-0.958456
```



« EXOTIC » PLOT COORDINATES AND THEIR USAGE

Safe Operating Area of a Mosfet (simple)

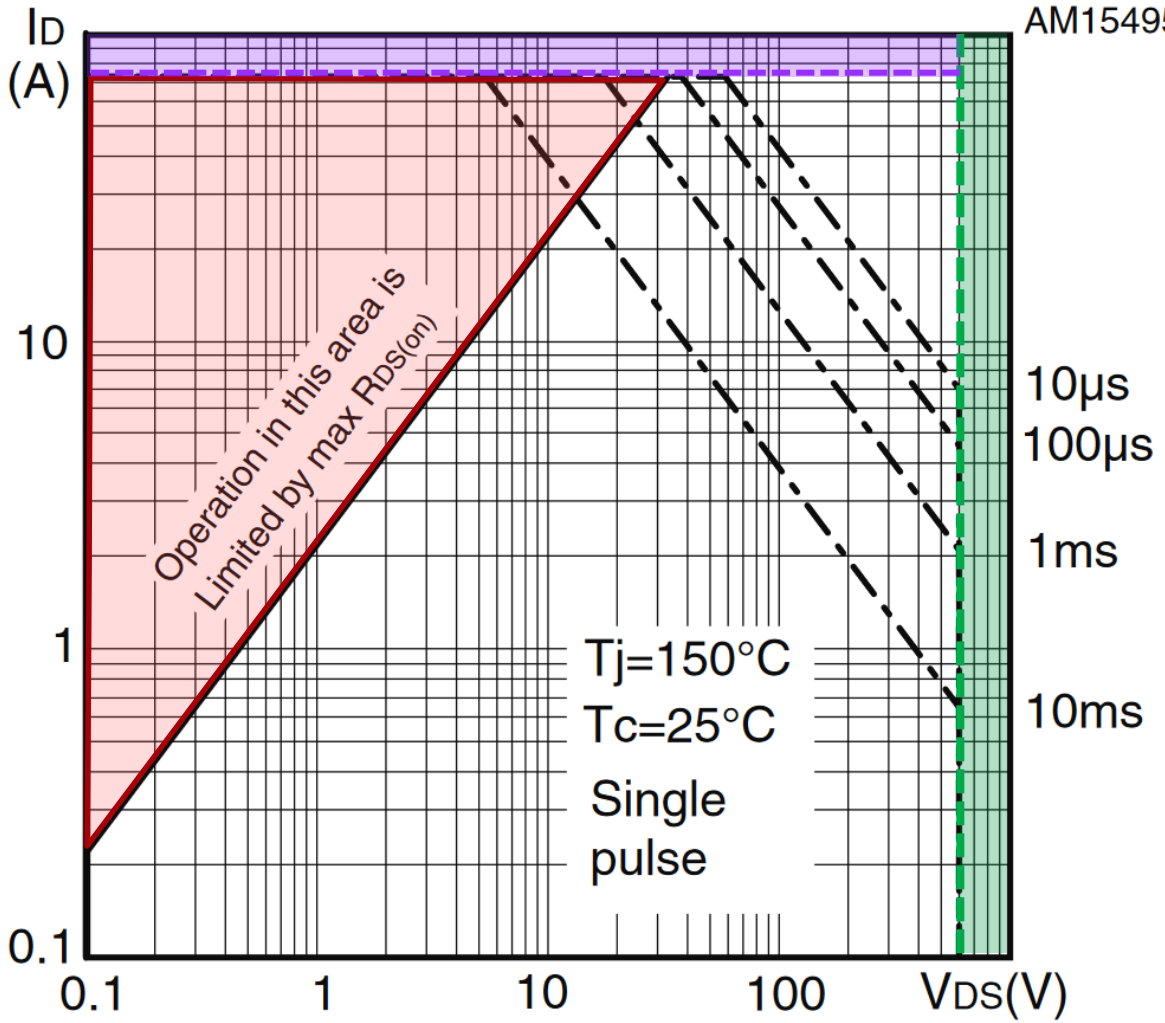


Table 5. On /off states

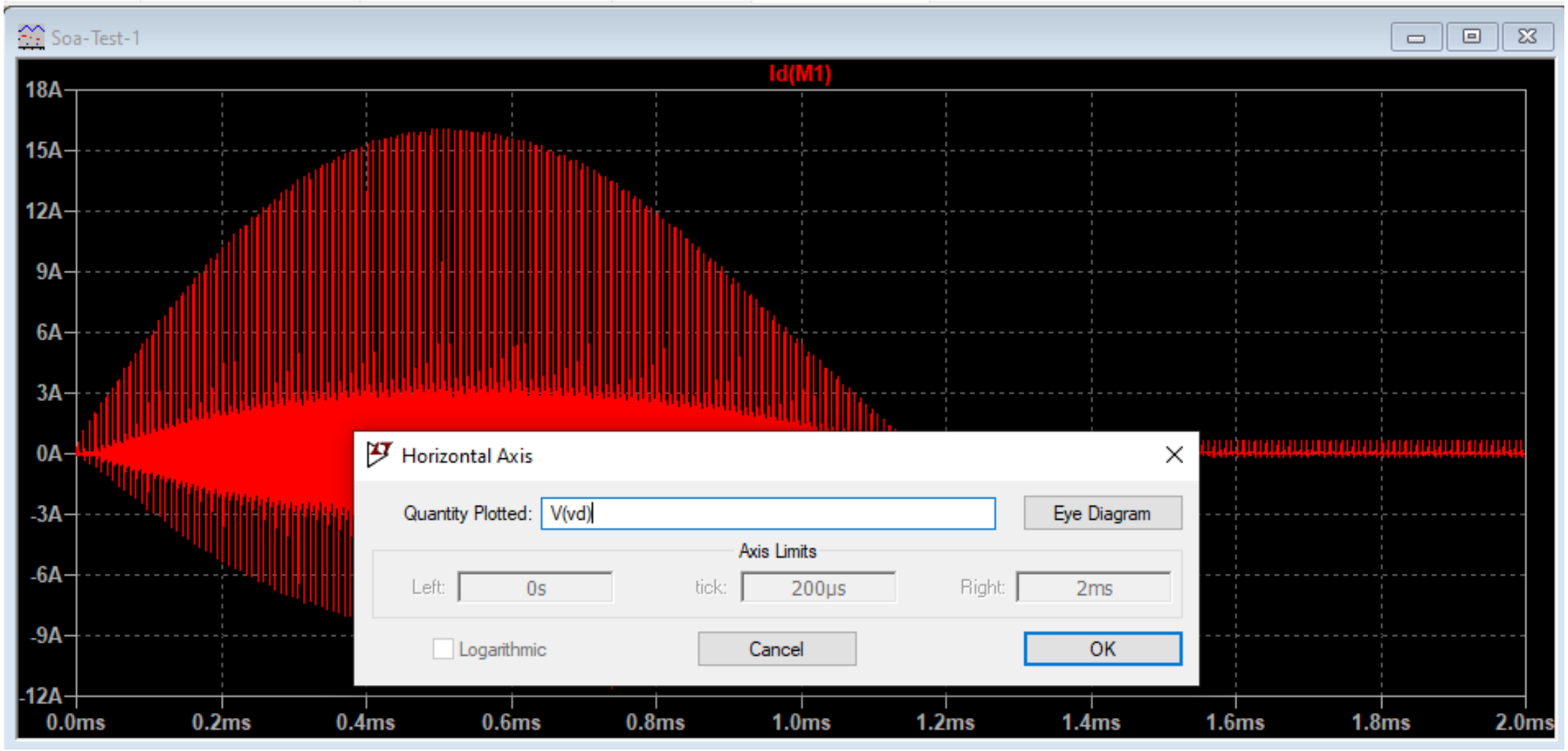
Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
$V_{(BR)DSS}$	Drain-source breakdown voltage	I_D = 1 mA, V_{GS} = 0	600			V
I_{DSS}	Zero gate voltage drain current (V_{GS} = 0)	V_{DS} = 600 V			1	µA
		V_{DS} = 600 V, T_C=125 °C			100	µA
I_{GSS}	Gate-body leakage current (V_{DS} = 0)	V_{GS} = ± 25 V			±10	µA
$V_{GS(th)}$	Gate threshold voltage	V_{DS} = V_{GS}, I_D = 250 µA	2	3	4	V
$R_{DS(on)}$	Static drain-source on-resistance	V_{GS} = 10 V, I_D = 9 A		0.168	0.19	Ω

Table 2. Absolute maximum ratings

Symbol	Parameter	Value	Unit
V_{GS}	Gate-source voltage	± 25	V
I_D	Drain current (continuous) at T_C = 25 °C	18	A
I_D	Drain current (continuous) at T_C = 100 °C	12	A
I_{DM} (1)	Drain current (pulsed)	72	A

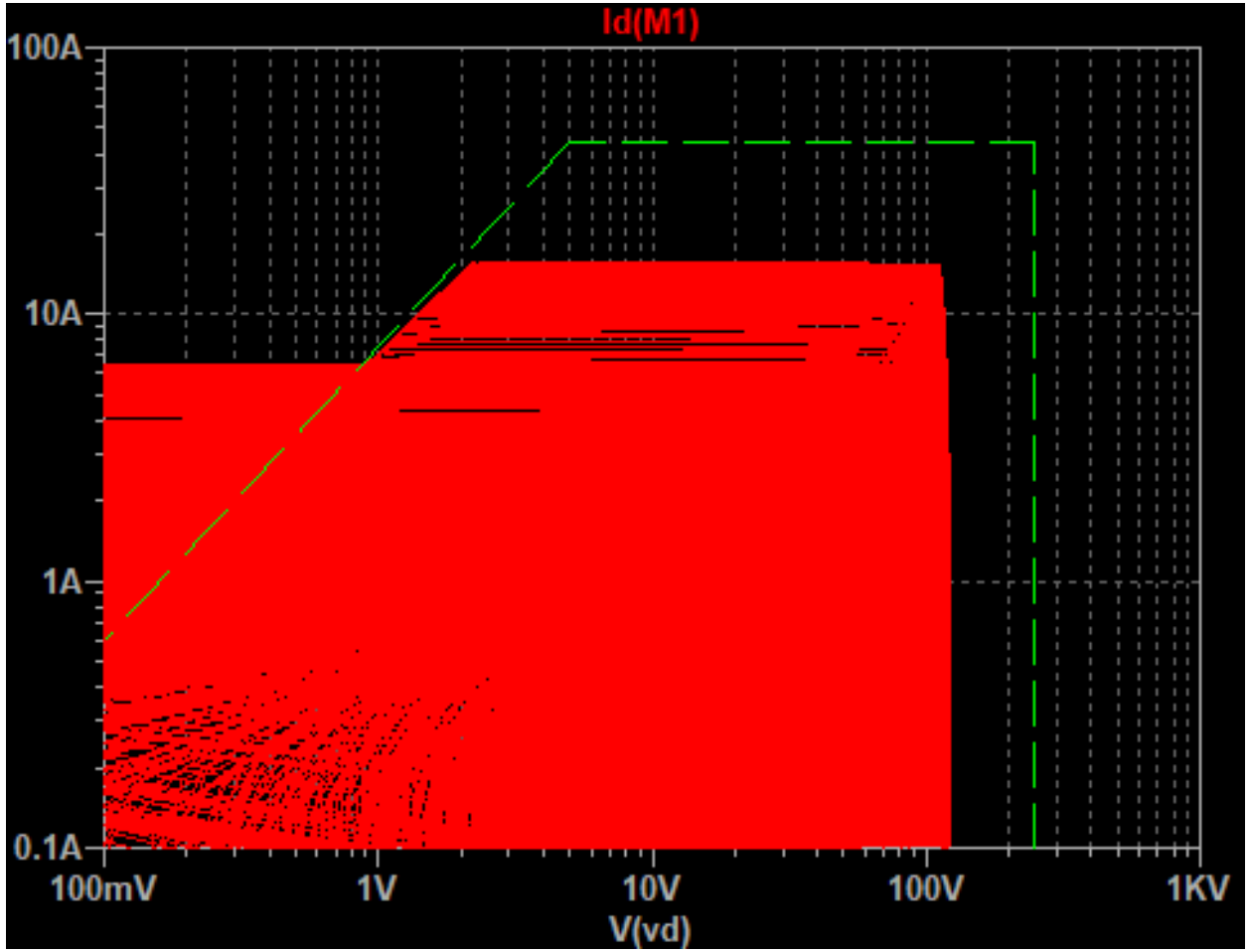
« EXOTIC » PLOT COORDINATES AND THEIR USAGE

Safe Operating Area of a Mosfet (simple)



« EXOTIC » PLOT COORDINATES AND THEIR USAGE

Safe Operating Area of a Mosfet (simple)



Soa-Test-1-edit.plt - Bloc-notes

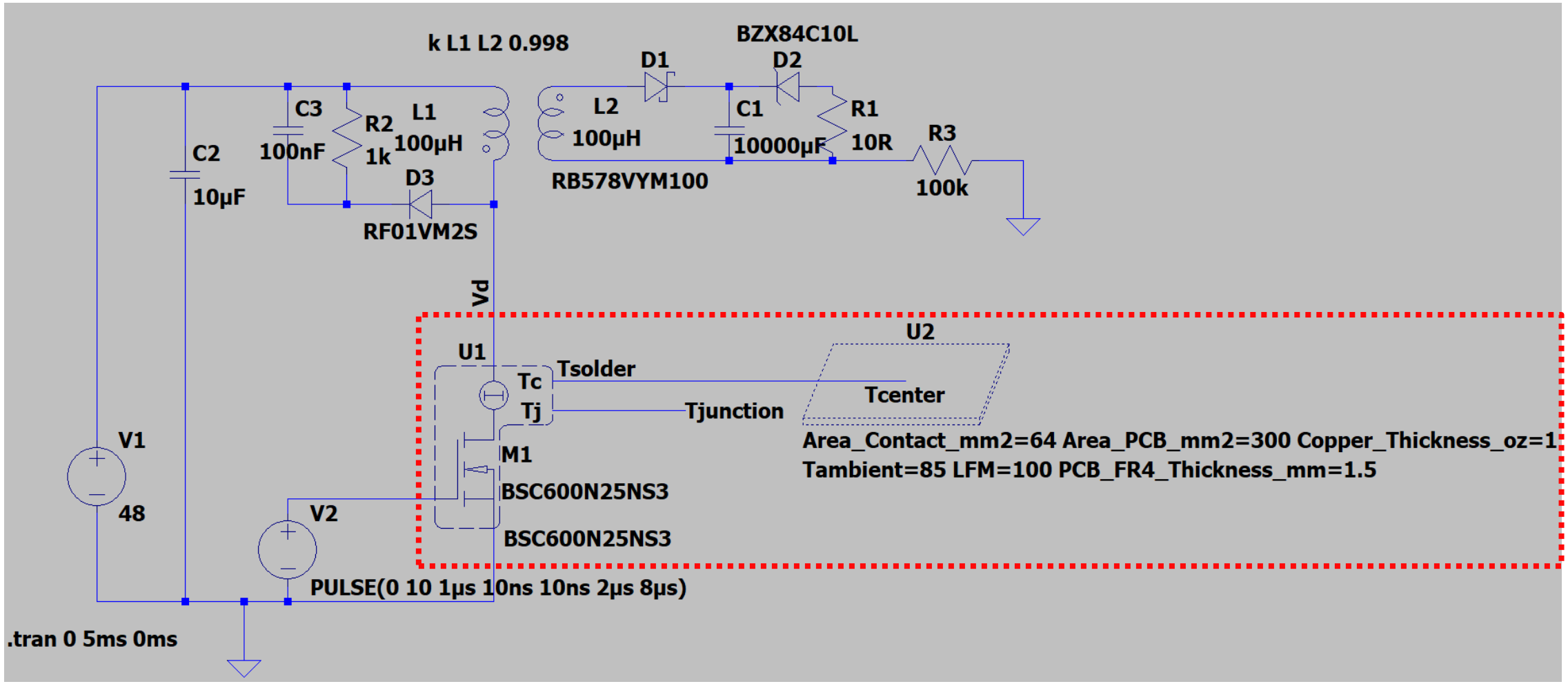
Fichier Edition Format Affichage Aide

```
[Transient Analysis]
{
   Npanes: 1
   {
      traces: 1 {34603012,0,"Id(M1)"}
      Parametric: "V(vd)"
      X: ('K',0,0.1,99.99,1000)
      Y[0]: ('_',0,0.1,9.99,100)
      Y[1]: (' ',0,1e+308,3,-1e+308)
      Amps: ('_',1,0,0,0.1,9.99,100)
      Log: 1 1 0
      GridStyle: 1
      Line: "A" 2 1 (250,0.100) (250,44)
      Line: "A" 2 1 (250,44) (5,44)
      Line: "A" 2 1 (5,44) (0.1,0.6)
   }
}
```

Ln 13, Col 1 | 100% | Windows (CRLF) | UTF-8

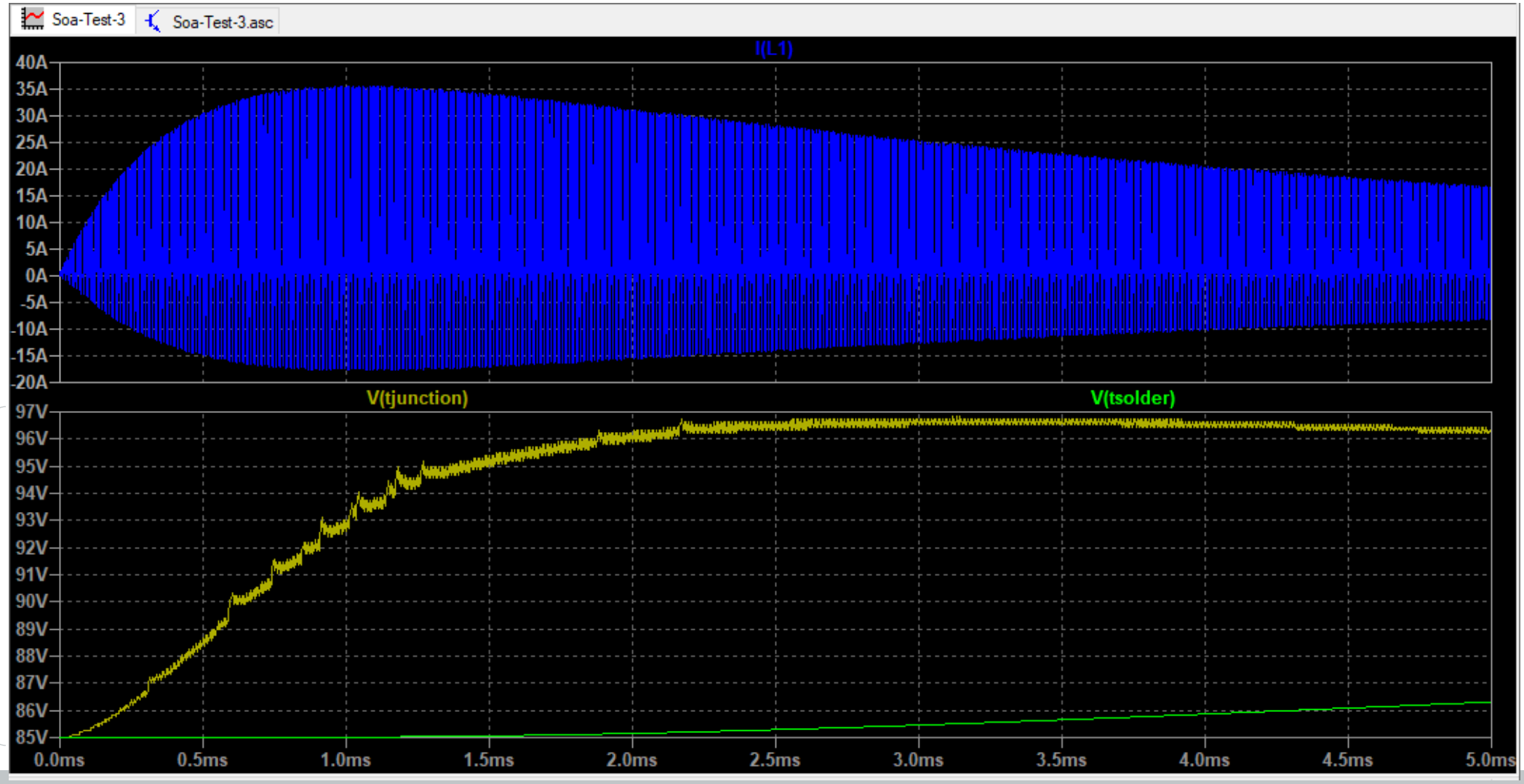
SOATHERM FUNCTIONS

Thermal simulations in LTSpice



SOATHERM FUNCTIONS

Thermal simulations in LTSpice

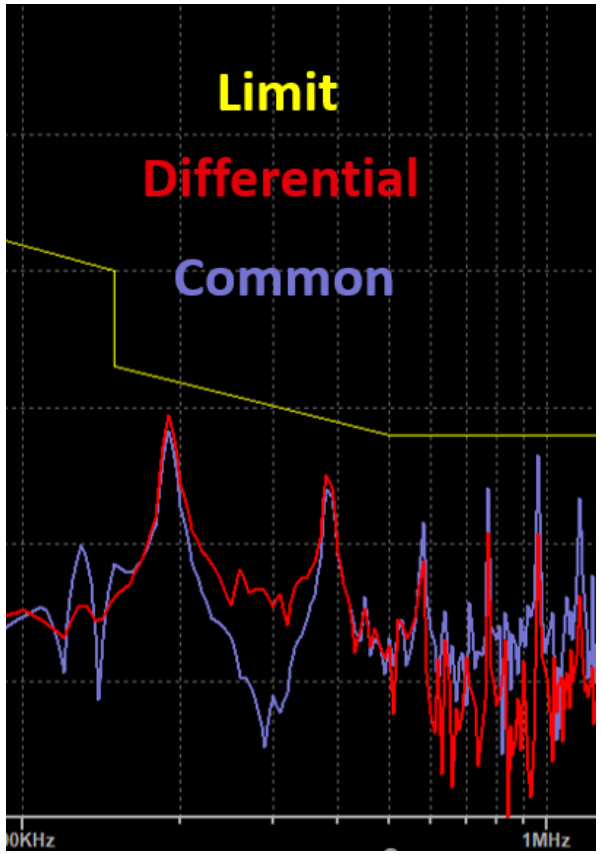


°C

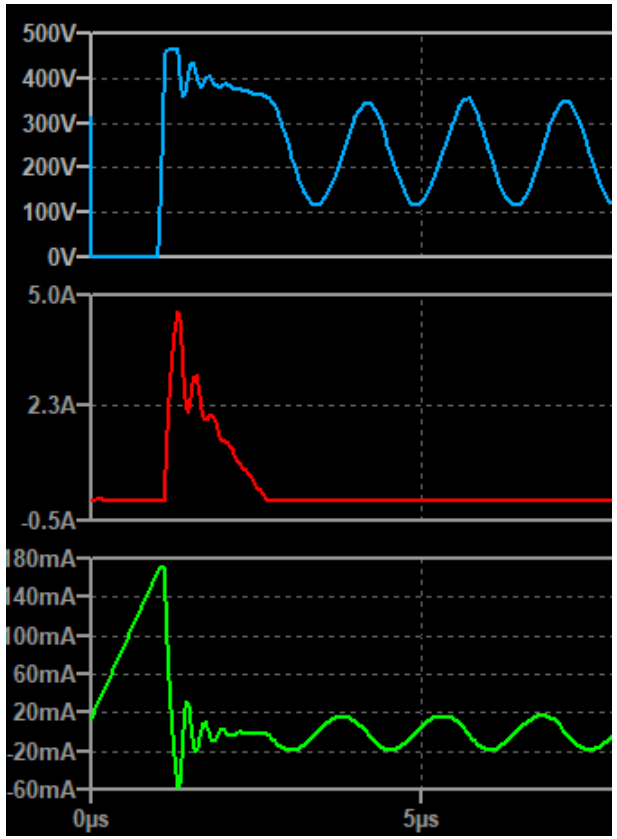
THANKS FOR WATCHING

Stay tuned ☺

- Anticipate EMC With LTSpice

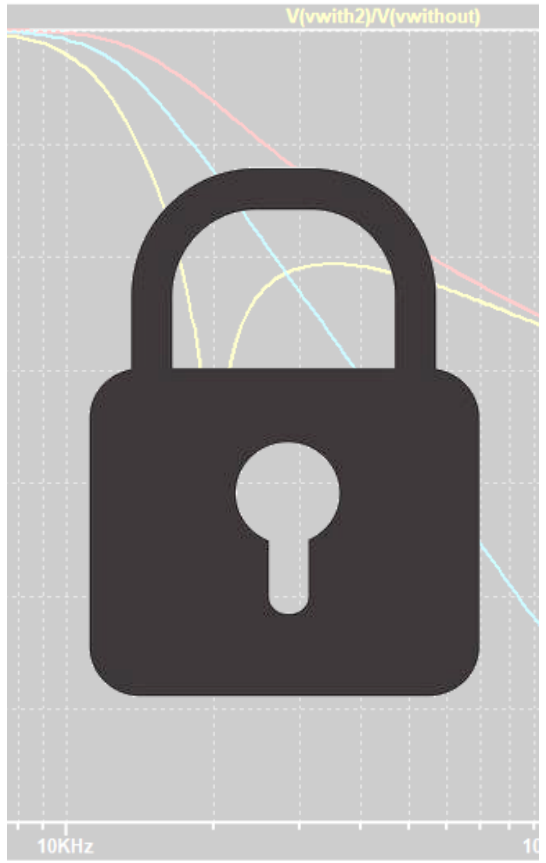


- Validation of power converters with LTSPICE (1/2)



- Signal Integrity With LTSpice

- LTSpice for EMC filter design



Questions

& Answers

We are here for you now!
Ask us directly via our chat or via E-Mail.

digital-we-days@we-online.com
Sylvain.LeBras@we-online.com