skidl\_2\_pyspice\_check

In [1]:

from skidl.pyspice import \*  
from PySpice.Spice.Netlist import Circuit

WARNING: KICAD\_SYMBOL\_DIR environment variable is missing, so the default KiCad symbol libraries won't be searched.

# Checking tool[¶](#Checking-tool)

In [2]:

def netlist\_comp\_check(skidl\_netlist, pyspice\_netlist):  
 """  
 Simple dumb check tool to compare the netlist from sckidl and pyspice   
   
 Args:  
 skidl\_netlist (PySpice.Spice.Netlist.Circuit): resulting netlist obj from  
 skidl using skidl's `generate\_netlist` utlity to compare to pyspice direct  
 creation  
   
 pyspice\_netlist (PySpice.Spice.Netlist.Circuit): circuit obj created directly in pyspice via  
 `PySpice.Spice.Netlist.Circuit` to compare it's netlist to skidl produced one  
   
 Returns:  
 if skidl\_netlist is longer then pyspice\_netlist will return string statment saying: 'skidl\_netlist is longer then pyspice\_netlist'  
   
 if skidl\_netlist is shorter then pyspice\_netlist will return string statment saying: 'skidl\_netlist is shorter then pyspice\_netlist'  
   
 if skidl\_netlist and pyspice\_netlist are equall and but there are diffrances then will print  
 message of thoes difrances(|1 indexed) and return a list of indexs where the skidl netlist is differs from the pyspice one  
   
 if skidl\_netlist == pyspice\_netlist then will return the word: 'Match'  
   
 TODO: Where should I start  
 """  
 #only care about the final netlist string  
 skidl\_netlist=skidl\_netlist.str()  
 pyspice\_netlist=pyspice\_netlist.str()  
   
 #check the lengths  
 if len(skidl\_netlist)>len(pyspice\_netlist):  
 return('skidl\_netlist is longer then pyspice\_netlist')  
 elif len(skidl\_netlist)<len(pyspice\_netlist):  
 return('skidl\_netlist is shorter then pyspice\_netlist')   
   
 #compare strings char by char  
 else:  
 string\_check=[i for i in range(len(skidl\_netlist)) if skidl\_netlist[i] != pyspice\_netlist[i]]  
 if string\_check==[]:  
 return 'Match'  
 else:  
 print('Match failed skidl\_netlist:')  
 print(f'{[i|1 for i in string\_check]}')  
 return string\_check

# Basic Elements[¶](#Basic-Elements)

## A | XSPICE code model (not checked)[¶](#A------------%7C-XSPICE-code-model-(not-checked))

PySpice/PySpice/Spice/BasicElement.py; (need to find):

skidl/skidl/libs/pyspice\_sklib.py; name="A"

## B | Behavioral (arbitrary) source (not checked)[¶](#B------------%7C-Behavioral-(arbitrary)-source-(not-checked))

PySpice/PySpice/Spice/BasicElement.py; class BehavioralSource:

skidl/skidl/libs/pyspice\_sklib.py; name="B"

ngspice 5.1: Bxxxx: Nonlinear dependent source (ASRC): BXXXXXXX n| n- <i=expr > <v=expr > <tc1=value > <tc2=value > <temp=value > <dtemp=value >

## C | Capacitor[¶](#C------------%7C-Capacitor)

PySpice/PySpice/Spice/BasicElement.py; class Capacitor(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="C"

ngspice 3.2.5 Capacitors:

CXXXXXXX n| n- <m=val> <scale=val> <temp=val> <dtemp=val> <tc1=val> <tc2=val> <ic=init\_condition >

### Notes[¶](#Notes)

In [3]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_C=C(ref='1', value=5, scale=5, temp=5, dtemp=5, ic=5, m=5)  
skidl\_C['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
C1 N1 N2 5 dtemp=5 ic=5 m=5 scale=5 temp=5

No errors or warnings found during netlist generation.

In [4]:

pyspice\_circ=Circuit('')  
pyspice\_circ.C('1', 'N1', 'N2', 5, scale=5, temp=5, dtemp=5, ic=5, m=5)  
print(pyspice\_circ)

.title   
C1 N1 N2 5 dtemp=5 ic=5 m=5 scale=5 temp=5

In [5]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[5]:

'Match'

## D | Diode[¶](#D------------%7C-Diode)

PySpice/PySpice/Spice/BasicElement.py; class Diode(FixedPinElement)

skidl/skidl/libs/pyspice\_sklib.py; name="D"

ngspice 7.1 Junction Diodes:

DXXXXXXX n| n- mname <area=val> <m=val> <pj=val> <ic=vd> <temp=val> <dtemp=val>

### Notes[¶](#Notes)

* ic: did not work in eather skidl or pyspice

In [6]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_D=D(ref='1',model=5, area=5, m=5, pj=5, off=5, temp=5, dtemp=5)  
skidl\_D['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

ERROR: Unknown SPICE model: 5

.title   
D1 N1 N2 5 area=5 dtemp=5 m=5 off pj=5 temp=5

0 warnings found during netlist generation.  
1 errors found during netlist generation.

In [7]:

pyspice\_circ=Circuit('')  
pyspice\_circ.D('1', 'N1', 'N2', model=5, area=5, m=5, pj=5, off=5, temp=5, dtemp=5)  
print(pyspice\_circ)

.title   
D1 N1 N2 5 area=5 dtemp=5 m=5 off pj=5 temp=5

In [8]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[8]:

'Match'

## E | Voltage-controlled voltage source (VCVS)[¶](#E------------%7C-Voltage-controlled-voltage-source-(VCVS))

PySpice/PySpice/Spice/BasicElement.py; class VoltageControlledVoltageSource(TwoPortElement)

skidl/skidl/libs/pyspice\_sklib.py; name="E"

ngspice 4.2.2 Exxxx: Linear Voltage-Controlled Voltage Sources (VCVS):

EXXXXXXX N| N- NC| NC- VALUE

### Notes[¶](#Notes)

In [9]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2'); net\_3=Net('N3'); net\_4=Net('N4')  
skidl\_E=E(ref='1', voltage\_gain=5)  
skidl\_E['ip', 'in']+=net\_1, net\_2; skidl\_E['op', 'on']+=net\_3, net\_4  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
E1 N3 N4 N1 N2 5

No errors or warnings found during netlist generation.

In [10]:

pyspice\_circ=Circuit('')  
pyspice\_circ.VoltageControlledVoltageSource('1', 'N3', 'N4', 'N1', 'N2', voltage\_gain=5)  
print(pyspice\_circ)

.title   
E1 N3 N4 N1 N2 5

In [11]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[11]:

'Match'

## F | Current-controlled current source (CCCs)[¶](#F------------%7C-Current-controlled-current-source-(CCCs))

PySpice/PySpice/Spice/BasicElement.py; class CurrentControlledCurrentSource(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="F"

ngspice 4.2.3 Fxxxx: Linear Current-Controlled Current Sources (CCCS):

FXXXXXXX N| N- VNAM VALUE <m=val>

### Notes[¶](#Notes)

In [12]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_F=F(ref='1', control='V1', current\_gain=5, m=5)  
skidl\_F['p', 'n']+=net\_1, net\_2;  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
F1 N1 N2 V1 5 m=5

No errors or warnings found during netlist generation.

In [13]:

pyspice\_circ=Circuit('')  
pyspice\_circ.CurrentControlledCurrentSource('1', 'N1', 'N2', 'V1', current\_gain=5, m=5)  
print(pyspice\_circ)

.title   
F1 N1 N2 V1 5 m=5

In [14]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[14]:

'Match'

## G | Voltage-controlled current source (VCCS)[¶](#G------------%7C-Voltage-controlled-current-source-(VCCS))

PySpice/PySpice/Spice/BasicElement.py; class VoltageControlledCurrentSource(TwoPortElement)

skidl/skidl/libs/pyspice\_sklib.py; name="G"

ngspice 4.2.1 Gxxxx: Linear Voltage-Controlled Current Sources (VCCS):

GXXXXXXX N| N- NC| NC- VALUE <m=val>

### Notes[¶](#Notes)

* 'transconductance' did not work in skidl; but gain did as did current\_gain

In [15]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2'); net\_3=Net('N3'); net\_4=Net('N4')  
skidl\_G=G(ref='1', current\_gain=5, m=5)  
skidl\_G['ip', 'in']+=net\_1, net\_2; skidl\_G['op', 'on']+=net\_3, net\_4  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
G1 N3 N4 N1 N2 5 m=5

In [16]:

pyspice\_circ=Circuit('')  
pyspice\_circ.VoltageControlledCurrentSource('1', 'N3', 'N4', 'N1', 'N2', transconductance=5, m=5)  
print(pyspice\_circ)

.title   
G1 N3 N4 N1 N2 5 m=5

In [17]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[17]:

'Match'

## H | Current-controlled voltage source (CCVS)[¶](#H------------%7C-Current-controlled-voltage-source-(CCVS))

PySpice/PySpice/Spice/BasicElement.py; class CurrentControlledVoltageSource(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="H"

ngspice 4.2.4 Hxxxx: Linear Current-Controlled Voltage Sources (CCVS):

HXXXXXXX n| n- vnam val

### Notes[¶](#Notes)

In [18]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_H=H(ref='1', control='V1', transresistance=5)  
skidl\_H['p', 'n']+=net\_1, net\_2;  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
H1 N1 N2 V1 5

No errors or warnings found during netlist generation.

In [19]:

pyspice\_circ=Circuit('')  
pyspice\_circ.CurrentControlledVoltageSource('1', 'N1', 'N2', 'V1', transresistance=5)  
print(pyspice\_circ)

.title   
H1 N1 N2 V1 5

In [20]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[20]:

'Match'

## I | Current source[¶](#I------------%7C-Current-source)

PySpice/PySpice/Spice/BasicElement.py; class CurrentSource(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="I"

ngspice 4.1 Independent Sources for Voltage or Current:

IYYYYYYY N| N- <

### Notes[¶](#Notes)

* a reduced version of ngspices IYYYYYYY only generating the arguement for < DC/TRAN VALUE >

In [21]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_I=I(ref='1', dc\_value=5)  
skidl\_I['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
I1 N1 N2 5

No errors or warnings found during netlist generation.

In [22]:

pyspice\_circ=Circuit('')  
pyspice\_circ.I('1', 'N1', 'N2', dc\_value=5)  
print(pyspice\_circ)

.title   
I1 N1 N2 5

In [23]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[23]:

'Match'

## J | Junction field effect transistor (JFET)[¶](#J------------%7C-Junction-field-effect-transistor-(JFET))

PySpice/PySpice/Spice/BasicElement.py; class JunctionFieldEffectTransistor(JfetElement)

skidl/skidl/libs/pyspice\_sklib.py; name="J"

ngspice 9.1 Junction Field-Effect Transistors (JFETs):

JXXXXXXX nd ng ns mname <ic=vds,vgs> <temp=t>

### Notes[¶](#Notes)

* ic: did not work in eather skidl or pyspice

In [24]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2'); net\_3=Net('N3')  
skidl\_J=J(ref='1',model=5, area=5, m=5, off=5, temp=5)  
skidl\_J['d', 'g', 's']+=net\_1, net\_2, net\_3  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

ERROR: Unknown SPICE model: 5

.title   
J1 N1 N2 N3 5 area=5 m=5 off temp=5

0 warnings found during netlist generation.  
1 errors found during netlist generation.

In [25]:

pyspice\_circ=Circuit('')  
pyspice\_circ.J('1', 'N1', 'N2', 'N3', model=5, area=5, m=5, off=5, temp=5)  
print(pyspice\_circ)

.title   
J1 N1 N2 N3 5 area=5 m=5 off temp=5

In [26]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[26]:

'Match'

## K | Coupled (Mutual) Inductors[¶](#K------------%7C-Coupled-(Mutual)-Inductors)

PySpice/PySpice/Spice/BasicElement.py; class CoupledInductor(AnyPinElement)

skidl/skidl/libs/pyspice\_sklib.py; name="K"

ngspice 3.2.11 Coupled (Mutual) Inductors:

KXXXXXXX LYYYYYYY LZZZZZZZ value

### Notes[¶](#Notes)

* need to get daves help on using K inside skidl
* the inductors must already exsist for pyspice to work

In [27]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_L1=L(ref='1', value=5, m=5, temp=5, dtemp=5, ic=5)  
skidl\_L1['p', 'n']+=net\_1, net\_2  
  
skidl\_L2=L(ref='2', value=5, m=5, temp=5, dtemp=5, ic=5)  
skidl\_L2['p', 'n']+=net\_1, net\_2  
  
#need to find out how to use this  
#skidl\_K=K()  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
L1 N1 N2 5 dtemp=5 ic=5 m=5 temp=5  
L2 N1 N2 5 dtemp=5 ic=5 m=5 temp=5

In [28]:

pyspice\_circ=Circuit('')  
#inductors need to exsist to then be coupled  
pyspice\_circ.L('1', 'N1', 'N2', 5, m=5, temp=5, dtemp=5, ic=5)  
pyspice\_circ.L('2', 'N1', 'N2', 5, m=5, temp=5, dtemp=5, ic=5)  
pyspice\_circ.K('1', 'L1', 'L2', coupling\_factor=5)  
print(pyspice\_circ)

.title   
L1 N1 N2 5 dtemp=5 ic=5 m=5 temp=5  
L2 N1 N2 5 dtemp=5 ic=5 m=5 temp=5  
K1 L1 L2 5

In [29]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[29]:

'skidl\_netlist is shorter then pyspice\_netlist'

## L | Inductor[¶](#L------------%7C-Inductor)

PySpice/PySpice/Spice/BasicElement.py; class Inductor(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="L"

ngspice 3.2.9 Inductors:

LYYYYYYY n| n- <nt=val> <m=val> <scale=val> <temp=val> <dtemp=val> <tc1=val> <tc2=val> <ic=init\_condition >

### Notes[¶](#Notes)

In [30]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_L=L(ref='1', value=5, m=5, temp=5, dtemp=5, ic=5)  
skidl\_L['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
L1 N1 N2 5 dtemp=5 ic=5 m=5 temp=5

In [31]:

pyspice\_circ=Circuit('')  
pyspice\_circ.L('1', 'N1', 'N2', 5, m=5, temp=5, dtemp=5, ic=5)  
print(pyspice\_circ)

.title   
L1 N1 N2 5 dtemp=5 ic=5 m=5 temp=5

In [32]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[32]:

'Match'

## M | Metal oxide field effect transistor (MOSFET)[¶](#M------------%7C-Metal-oxide-field-effect-transistor-(MOSFET))

PySpice/PySpice/Spice/BasicElement.py; class Mosfet(FixedPinElement)

skidl/skidl/libs/pyspice\_sklib.py; name="M"

ngspice 11.1 MOSFET devices:

MXXXXXXX nd ng ns nb mname <m=val> <l=val> <w=val> <ad=val> <as=val> <pd=val> <ps=val> <nrd=val> <nrs=val> <ic=vds, vgs, vbs> <temp=t>

### Notes[¶](#Notes)

* ic: did not work in eather skidl or pyspice

In [33]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2'); net\_3=Net('N3'); net\_4=Net('N4')  
skidl\_M=M(ref='1', model=5, m=5, l=5, w=5,   
 drain\_area=5, source\_area=5, drain\_perimeter=5, source\_perimeter=5,   
 drain\_number\_square=5, source\_number\_square=5,  
 off=5, temp=5)  
  
skidl\_M['d', 'g', 's', 'b']+=net\_1, net\_2, net\_3, net\_4  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

ERROR: Unknown SPICE model: 5  
  
0 warnings found during netlist generation.  
1 errors found during netlist generation.

.title   
M1 N1 N2 N3 N4 5 ad=5 nrd=5 pd=5 l=5 m=5 off as=5 nrs=5 ps=5 temp=5 w=5

In [34]:

pyspice\_circ=Circuit('')  
pyspice\_circ.M('1', 'N1', 'N2', 'N3', 'N4', model=5, m=5, l=5, w=5,   
 drain\_area=5, source\_area=5, drain\_perimeter=5, source\_perimeter=5,   
 drain\_number\_square=5, source\_number\_square=5,  
 off=5, temp=5)  
print(pyspice\_circ)

.title   
M1 N1 N2 N3 N4 5 ad=5 nrd=5 pd=5 l=5 m=5 off as=5 nrs=5 ps=5 temp=5 w=5

In [35]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[35]:

'Match'

| N | Numerical device for GSS |

| O | Lossy transmission line |

| P | Coupled multiconductor line (CPL) |

## Q | Bipolar junction transistor (BJT)[¶](#Q------------%7C-Bipolar-junction-transistor-(BJT))

PySpice/PySpice/Spice/BasicElement.py; class BipolarJunctionTransistor(FixedPinElement)

skidl/skidl/libs/pyspice\_sklib.py; name="Q"

ngspice 8.1 Bipolar Junction Transistors (BJTs):

QXXXXXXX nc nb ne mname <area=val> <areac=val> <areab=val> <m=val> <ic=vbe,vce> <temp=val> <dtemp=val>

### Notes[¶](#Notes)

* could not get the substrate connection working in pyspice but it worked fine with skidl
* ic: did not work in eather skidl or pyspice

In [36]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2'); net\_3=Net('N3'); net\_4=Net('N4')  
skidl\_Q=Q(ref='1',model=5,   
 area=5, areab=5, areac=5,  
 m=5, off=5, temp=5, dtemp=5)  
skidl\_Q['c', 'b', 'e']+=net\_1, net\_2, net\_3  
  
#skidl will make the substrate connection fine but could not get pyspice to do so  
#therefore skiping for the time being  
#skidl\_Q['s']+=net\_4  
  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

ERROR: Unknown SPICE model: 5  
  
0 warnings found during netlist generation.  
1 errors found during netlist generation.

.title   
Q1 N1 N2 N3 5 area=5 areab=5 areac=5 dtemp=5 m=5 off temp=5

In [37]:

pyspice\_circ=Circuit('')  
pyspice\_circ.Q('1', 'N1', 'N2', 'N3', model=5, area=5, areab=5, areac=5,  
 m=5, off=5, temp=5, dtemp=5,   
 #could not get the substrate connection working in pyspice  
 #ns='N4'  
 )  
   
print(pyspice\_circ)

.title   
Q1 N1 N2 N3 5 area=5 areab=5 areac=5 dtemp=5 m=5 off temp=5

In [38]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[38]:

'Match'

## R | Resistor[¶](#R------------%7C-Resistor)

PySpice/PySpice/Spice/BasicElement.py; class Resistor(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="R"

ngspice 3.2.1 Resistors:

RXXXXXXX n| n- <resistance|r=>value <ac=val> <m=val> <scale=val> <temp=val> <dtemp=val> <tc1=val> <tc2=val> <noisy=0|1>

### Notes[¶](#Notes)

In [39]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_R=R(ref='1', value=5, ac=5, m=5, scale=5, temp=5, dtemp=5, noisy=1)  
skidl\_R['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
R1 N1 N2 5 ac=5 dtemp=5 m=5 noisy=1 scale=5 temp=5

In [40]:

pyspice\_circ=Circuit('')  
pyspice\_circ.R('1', 'N1', 'N2', 5, ac=5, m=5, scale=5, temp=5, dtemp=5, noisy=1)  
print(pyspice\_circ)

.title   
R1 N1 N2 5 ac=5 dtemp=5 m=5 noisy=1 scale=5 temp=5

In [41]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[41]:

'Match'

| S | Switch (voltage-controlled) |

| T | Lossless transmission line |

| U | Uniformly distributed RC line |

## V | Voltage source[¶](#V-%7C-Voltage-source)

PySpice/PySpice/Spice/BasicElement.py; class VoltageSource(DipoleElement)

skidl/skidl/libs/pyspice\_sklib.py; name="V"

ngspice 4.1 Independent Sources for Voltage or Current:

VXXXXXXX N| N- < DC/TRAN VALUE >

### Notes[¶](#Notes)

* a reduced version of ngspices VXXXXXXX only generating the arguement for < DC/TRAN VALUE >

In [42]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_V=V(ref='1', dc\_value=5)  
skidl\_V['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
V1 N1 N2 5

In [43]:

pyspice\_circ=Circuit('')  
pyspice\_circ.V('1', 'N1', 'N2', dc\_value=5)  
print(pyspice\_circ)

.title   
V1 N1 N2 5

In [44]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[44]:

'Match'

| W | Switch (current-controlled) |

| X | Subcircuit |

| Y | Single lossy transmission line (TXL) |

| Z | Metal semiconductor field effect transistor (MESFET) |

## Z | Metal semiconductor field effect transistor (MESFET)[¶](#Z------------%7C-Metal-semiconductor-field-effect-transistor-(MESFET))

PySpice/PySpice/Spice/BasicElement.py; class Mesfet(JfetElement)

skidl/skidl/libs/pyspice\_sklib.py; name="Z"

ngspice 10.1 MESFETs:

ZXXXXXXX ND NG NS MNAME <IC=VDS, VGS>

### Notes[¶](#Notes)

* ic: did not work in eather skidl or pyspice

In [45]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2'); net\_3=Net('N3')  
skidl\_Z=Z(ref='1',model=5, area=5, m=5, off=5)  
skidl\_Z['d', 'g', 's']+=net\_1, net\_2, net\_3  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

ERROR: Unknown SPICE model: 5

.title   
Z1 N1 N2 N3 5 area=5 m=5 off

0 warnings found during netlist generation.  
1 errors found during netlist generation.

In [46]:

pyspice\_circ=Circuit('')  
pyspice\_circ.Z('1', 'N1', 'N2', 'N3', model=5, area=5, m=5, off=5)  
print(pyspice\_circ)

.title   
Z1 N1 N2 N3 5 area=5 m=5 off

In [47]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[47]:

'Match'

# Highlevel Elements SinusoidalMixin Based[¶](#Highlevel-Elements-SinusoidalMixin-Based)

## Note in Armour's fort added as\_phase[¶](#Note-in-Armour's-fort-added-as_phase)

SinusoidalMixin is the base translation class for sinusoid wave waveform sources, in other words even thou ngspice compines most sinusoid source as just argument extations to exsisting DC source to create AC souces through pyspice to ngspice these elements must be used

## SinusoidalMixin args:[¶](#SinusoidalMixin-args:)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Parameter | Default Value | Units |
| Vo | offset |  | V, A |
| ------ | ---------------- | --------------- | ------- |
| Va | amplitude |  | V, A |
| ------ | ---------------- | --------------- | ------- |
| f | frequency | 1 / TStop | Hz |
| ------ | ---------------- | --------------- | ------- |
| Td | delay | 0.0 | sec |
| ------ | ---------------- | --------------- | ------- |
| Df | damping factor | 0.01 | 1/sec |
| ------ | ---------------- | --------------- | ------- |

so for a AC SIN voltage sours it's output should be equilint to the following:

$$V(t) = \begin{cases} V\_o & \text{if}\ 0 \leq t < T\_d, \\ V\_o + V\_a e^{-D\_f(t-T\_d)} \sin\left(2\pi f (t-T\_d)\right) & \text{if}\ T\_d \leq t < T\_{stop}. \end{cases}$$

## SinusoidalVoltageSource (AC)[¶](#SinusoidalVoltageSource-(AC))

PySpice/PySpice/Spice/HighLevelElement.py; class SinusoidalVoltageSource(VoltageSource, VoltageSourceMixinAbc, SinusoidalMixin)

skidl/skidl/libs/pyspice\_sklib.py; name="SINEV"

ngspice 4.1 Independent Sources for Voltage or Current & 4.1.2 Sinusoidal:

VXXXXXXX N+ N- < DC/TRAN VALUE > <AC <ACMAG >> <DISTOF1 <F1MAG >> <DISTOF2 <F2MAG >>

SIN(VO VA FREQ TD THETA PHASE)

### Notes[¶](#Notes)

* a amalgumation of ngspice's Independent Sources for Voltage & Sinusoidal statment for transint simulations

In [48]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_SINV=SINEV(ref='1',   
 #transit sim statments  
 offset=5,amplitude=5, frequency=5 , delay=5, damping\_factor=5,  
 #ac sim statments  
 ac\_magnitude=5, dc\_offset=5)  
  
skidl\_SINV['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
V1 N1 N2 DC 5V AC 5V SIN(5V 5V 5Hz 5s 5Hz)

In [49]:

pyspice\_circ=Circuit('')  
pyspice\_circ.SinusoidalVoltageSource('1', 'N1', 'N2',   
 #transit sim statments  
 offset=5,amplitude=5, frequency=5 , delay=5, damping\_factor=5,  
 #ac sim statments  
 ac\_magnitude=5, dc\_offset=5  
   
 )  
print(pyspice\_circ)

.title   
V1 N1 N2 DC 5V AC 5V SIN(5V 5V 5Hz 5s 5Hz)

In [50]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[50]:

'Match'

## SinusoidalCurrentSource (AC)[¶](#SinusoidalCurrentSource-(AC))

PySpice/PySpice/Spice/HighLevelElement.py; class class SinusoidalCurrentSource(CurrentSource, CurrentSourceMixinAbc, SinusoidalMixin):

skidl/skidl/libs/pyspice\_sklib.py; name="SINEI"

ngspice 4.1 Independent Sources for Voltage or Current & 4.1.2 Sinusoidal:

IYYYYYYY N+ N- < DC/TRAN VALUE > <AC <ACMAG >> <DISTOF1 <F1MAG >> <DISTOF2 <F2MAG >>

SIN(VO VA FREQ TD THETA PHASE)

### Notes[¶](#Notes)

* a amalgumation of ngspice's Independent Sources for Voltage & Sinusoidal statment for transint simulations

In [51]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_SINI=SINEI(ref='1',   
 #transit sim statments  
 offset=5,amplitude=5, frequency=5 , delay=5, damping\_factor=5,  
 #ac sim statments  
 ac\_magnitude=5, dc\_offset=5)  
  
skidl\_SINI['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

No errors or warnings found during netlist generation.

.title   
I1 N1 N2 DC 5A AC 5A SIN(5A 5A 5Hz 5s 5Hz)

In [52]:

pyspice\_circ=Circuit('')  
pyspice\_circ.SinusoidalCurrentSource('1', 'N1', 'N2',   
 #transit sim statments  
 offset=5,amplitude=5, frequency=5 , delay=5, damping\_factor=5,  
 #ac sim statments  
 ac\_magnitude=5, dc\_offset=5  
   
 )  
print(pyspice\_circ)

.title   
I1 N1 N2 DC 5A AC 5A SIN(5A 5A 5Hz 5s 5Hz)

In [53]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[53]:

'Match'

## AcLine(SinusoidalVoltageSource)[¶](#AcLine(SinusoidalVoltageSource))

PySpice/PySpice/Spice/HighLevelElement.py; class AcLine(SinusoidalVoltageSource)

skidl/skidl/libs/pyspice\_sklib.py; NOT IMPLIMENTED

ngspice 4.1 Independent Sources for Voltage or Current:

VXXXXXXX N+ N- < DC/TRAN VALUE > <AC <ACMAG >> <DISTOF1 <F1MAG >> <DISTOF2 <F2MAG >>

### Notes[¶](#Notes)

* it's a pyspice only wraper around pyspices SinusoidalVoltageSource that makes a pure for transisint simulation only SIN voltage source with the only arguments being rms\_voltage and frequency
* pyspice does the rms to amplitute conversion internaly
* pyspice does not have a offset arg
* pyspice does not have a delay arg
* pyspice does not have a damping\_factor arg
* pyspice does not have a ac\_magnitude arg
* pyspice does not have a dc\_offset arg
* pspice still gives a AC output of the default 1V; this needs to be changed to be equal to amplitude internal value or else will give aid in producing incorect results with **ac** simulations

In [54]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
# Skidle does not impliment an AcLine equivlent at this time  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title

No errors or warnings found during netlist generation.

In [55]:

pyspice\_circ=Circuit('')  
pyspice\_circ.AcLine('1', 'N1', 'N2',   
 #transit sim statments  
 rms\_voltage=8, frequency=5   
   
 )  
print(pyspice\_circ)

.title   
V1 N1 N2 DC 0V AC 1V SIN(0V 11.313708498984761V 5Hz 0s 0Hz)

In [56]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[56]:

'skidl\_netlist is shorter then pyspice\_netlist'

# Highlevel Elements PulseMixin Based[¶](#Highlevel-Elements-PulseMixin-Based)

# Highlevel Elements ExponentialMixin Based[¶](#Highlevel-Elements-ExponentialMixin-Based)

ExponentialMixin is the base translation class for exponential shped sources used for transisint simulations. Typicly used for simulating responce to charing and discharing events from capcitor/inductor networks. Pyspice does not include ac arguements that are technicly allowed by ngspice

## ExponentialMixin args:[¶](#ExponentialMixin-args:)

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Parameter | Default Value | Units |
| V1 | Initial value |  | V, A |
| ------ | -------------------- | --------------- | ------- |
| V2 | pulsed value |  | V, A |
| ------ | -------------------- | --------------- | ------- |
| Td1 | rise delay time | 0.0 | sec |
| ------ | -------------------- | --------------- | ------- |
| tau1 | rise time constant | Tstep | sec |
| ------ | -------------------- | --------------- | ------- |
| Td2 | fall delay time | Td1 | Tstep |
| ------ | -------------------- | --------------- | ------- |
| tau2 | fall time constant | Tstep | sec |
| ------ | -------------------- | --------------- | ------- |

so for a expoential based voltage source it's output should be equilint to the following:

$$V(t) = \begin{cases} V\_1 & \text{if}\ 0 \leq t < T\_{d1}, \\ V\_1 + V\_{21} ( 1 − e^{-\frac{t-T\_{d1}}{\tau\_1}} ) & \text{if}\ T\_{d1} \leq t < T\_{d2}, \\ V\_1 + V\_{21} ( 1 − e^{-\frac{t-T\_{d1}}{\tau\_1}} ) + V\_{12} ( 1 − e^{-\frac{t-T\_{d2}}{\tau\_2}} ) & \text{if}\ T\_{d2} \leq t < T\_{stop} \end{cases}$$

where $V\_{21} = V\_2 - V\_1$ and $V\_{12} = V\_1 - V\_2$

## ExponentialVoltageSource[¶](#ExponentialVoltageSource)

PySpice/PySpice/Spice/HighLevelElement.py; class ExponentialVoltageSource(VoltageSource, VoltageSourceMixinAbc, ExponentialMixin)

skidl/skidl/libs/pyspice\_sklib.py; name="EXPV"

ngspice 4.1 Independent Sources for Voltage or Current & 4.1.3 Exponential:

VXXXXXXX N+ N-

EXP(V1 V2 TD1 TAU1 TD2 TAU2)

### Notes[¶](#Notes)

* should technicly also alow dc and ac values from ngspice Independent voltage source statment

In [57]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_EXPV=EXPV(ref='1',   
 #transit sim statments  
 initial\_value=5,pulsed\_value=5, rise\_delay\_time=5 , rise\_time\_constant=5, fall\_delay\_time=5, fall\_time\_constant=5,  
 )  
  
skidl\_EXPV['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
V1 N1 N2 EXP(5V 5V 5s 5s 5s 5s)

No errors or warnings found during netlist generation.

In [58]:

pyspice\_circ=Circuit('')  
pyspice\_circ.ExponentialVoltageSource('1', 'N1', 'N2',   
 #transit sim statments  
 initial\_value=5,pulsed\_value=5, rise\_delay\_time=5 , rise\_time\_constant=5, fall\_delay\_time=5, fall\_time\_constant=5,  
   
 )  
print(pyspice\_circ)

.title   
V1 N1 N2 EXP(5V 5V 5s 5s 5s 5s)

In [59]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[59]:

'Match'

## ExponentialCurrentSource[¶](#ExponentialCurrentSource)

PySpice/PySpice/Spice/HighLevelElement.py; class ExponentialCurrentSource(VoltageSource, VoltageSourceMixinAbc, ExponentialMixin)

skidl/skidl/libs/pyspice\_sklib.py; name="EXPI"

ngspice 4.1 Independent Sources for Voltage or Current & 4.1.3 Exponential:

IXXXXXXX N+ N-

EXP(I1 I2 TD1 TAU1 TD2 TAU2)

### Notes[¶](#Notes)

* should technicly also alow dc and ac values from ngspice Independent voltage source statment

In [60]:

reset()  
net\_1=Net('N1'); net\_2=Net('N2')  
skidl\_EXPI=EXPI(ref='1',   
 #transit sim statments  
 initial\_value=5,pulsed\_value=5, rise\_delay\_time=5 , rise\_time\_constant=5, fall\_delay\_time=5, fall\_time\_constant=5,  
 )  
  
skidl\_EXPI['p', 'n']+=net\_1, net\_2  
skidl\_circ=generate\_netlist()  
print(skidl\_circ)

.title   
I1 N1 N2 EXP(5A 5A 5s 5s 5s 5s)

No errors or warnings found during netlist generation.

In [61]:

pyspice\_circ=Circuit('')  
pyspice\_circ.ExponentialCurrentSource('1', 'N1', 'N2',   
 #transit sim statments  
 initial\_value=5,pulsed\_value=5, rise\_delay\_time=5 , rise\_time\_constant=5, fall\_delay\_time=5, fall\_time\_constant=5,  
   
 )  
print(pyspice\_circ)

.title   
I1 N1 N2 EXP(5A 5A 5s 5s 5s 5s)

In [62]:

netlist\_comp\_check(skidl\_circ, pyspice\_circ)

Out[62]:

'Match'

# Highlevel Elements PieceWiseLinearMixin Based[¶](#Highlevel-Elements-PieceWiseLinearMixin-Based)

# Highlevel Elements SingleFrequencyFMMixin Based[¶](#Highlevel-Elements-SingleFrequencyFMMixin-Based)

# Highlevel Elements AmplitudeModulatedMixin Based[¶](#Highlevel-Elements-AmplitudeModulatedMixin-Based)

# Highlevel Elements RandomMixin Based[¶](#Highlevel-Elements-RandomMixin-Based)

In [ ]: