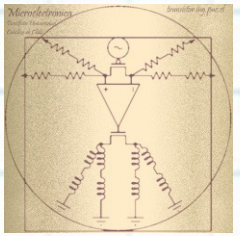


Introducción a LTspice

Angel Abusleme
Pontificia Universidad Católica de Chile
angel@ing.puc.cl
http://transistor.ing.puc.cl/

Índice

- Introducción a SPICE
- Primeros pasos con LTspice
- Hilando un poco más fino…
- Más información

Introducción a SPICE

```
M2 N001 Vtailp Vdd Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)'
M3 N012 Vm N001 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m='Minput'
M4 N011 Vp N001 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m='Minput'
M5 N007 N007 N001 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m='6*Minput'
M9 N005 Vp N014 Vss nch.7 l='L' w='MN*lambda' ad='2*HDIFP*(MN*lambda+DXW)' as='2*HDIFP*(MN*lambda+DXW)' pd='4*HDIFP+2*(MN*lambda+DXW)' ps='4*HDIFP+2*(MN*lambda+DXW)' m='Minput'
M11 N005 Vlp Vdd Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)'
M12 N002 Vcp N005 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)'
M13 N004 Vlp Vdd Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)'
M14 N008 Vcp N004 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)'
M15 N012 N008 Vss Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)'
M16 N011 N008 Vss Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)'
M17 N009 Vcn N011 Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)'
M18 N008 Vcn N012 Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)'
M29 N003 N003 Vdd Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m=0.1
M30 N006 N006 N003 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m=0.1
M31 N010 Vlp Vdd Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m=0.1
M32 N013 N013 Vss Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)' m=0.1
M33 N010 N010 N013 Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)' m=0.1
M34 N006 Vln Vss Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)' m=0.1
M35 N002 N010 N009 Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)' m=1
M36 Vout N009 Vss Vss nch.7 l='L' w='64*lambda' ad='2*HDIFP*(64*lambda+DXW)' as='2*HDIFP*(64*lambda+DXW)' pd='4*HDIFP+2*(64*lambda+DXW)' ps='4*HDIFP+2*(64*lambda+DXW)' m='mout'
M37 N009 N006 N002 Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m=1
M38 Vout N002 Vdd Vdd pch.7 l='L' w='360*lambda' ad='2*HDIFP*(360*lambda+DXW)' as='2*HDIFP*(360*lambda+DXW)' pd='4*HDIFP+2*(360*lambda+DXW)' ps='4*HDIFP+2*(360*lambda+DXW)' m='mout'
R1 N002 P002 'Rz'
R2 P001 N009 'Rz'
M1 N004 Vm N014 Vss nch.7 l='L' w='MN*lambda' ad='2*HDIFP*(MN*lambda+DXW)' as='2*HDIFP*(MN*lambda+DXW)' pd='4*HDIFP+2*(MN*lambda+DXW)' ps='4*HDIFP+2*(MN*lambda+DXW)' m='Minput'
M6 N007 N007 N014 Vss nch.7 l='L' w='MN*lambda' ad='2*HDIFP*(MN*lambda+DXW)' as='2*HDIFP*(MN*lambda+DXW)' pd='4*HDIFP+2*(MN*lambda+DXW)' ps='4*HDIFP+2*(MN*lambda+DXW)' m='6*Minput'
M7 N014 Vtailn Vss Vss nch.7 l='L' w='MN*lambda' ad='2*HDIFP*(MN*lambda+DXW)' as='2*HDIFP*(MN*lambda+DXW)' pd='4*HDIFP+2*(MN*lambda+DXW)' ps='4*HDIFP+2*(MN*lambda+DXW)'
.model NMOS NMOS
.model PMOS PMOS
.lib C:\Program Files (x86)\LTC\LTspiceIV\lib\cmp\standard.mos
```

Contexto de SPICE

- SPICE es típicamente parte de una *Suite EDA*
- ¿Qué significa EDA?
  - Los programas para diseño electrónico se denominan de forma genérica *Software de Automatización de Diseño Electrónico*, o *EDA* (por su sigla en inglés)
  - Una suite EDA incluye programas de:
    - captura (edición) de esquemáticos,
    - simulación (SPICE),
    - visualización de formas de ondas,
    - simulación de circuitos cuasi estáticos,
    - layout o diseño a nivel de topología física de un circuito integrado,
    - verificación (DRC, LVS, antena), para comparar layout con esquemático,
    - extracción de componentes,
    - extracción de elementos parásitos,
    - diseño de PCB,
    - etc.

Contexto de SPICE

- ¿Cuáles son las principales compañías de EDA?
  - Synopsys
  - Cadence
  - Mentor Graphics
  - Tanner
  - Zuken
  - Simucad
  - Magma Design Automation
  - etc. (ver http://en.wikipedia.org/wiki/List_of_EDA_companies)
- ¿Todas las suite EDA usan el mismo SPICE?
  - ¡No! Cada suite EDA tiene su propio "sabor" de SPICE
    - De un abecedario de denominaciones
  - También hay programas de SPICE independientes
    - ¡¡¡e incluso gratis!!!
  - Todos los SPICE tienen la misma estructura base
    - y sintaxis parecida

Contexto de SPICE

- ¿Para qué sirve SPICE?
  - SPICE sirve para simular el comportamiento de circuitos electrónicos
    - SPICE predice con gran precisión el desempeño de un circuito
    - Permite ahorrar muchísimo tiempo y recursos
      - es más barato probar un circuito en el computador que en proto, placa o silicio

    … entonces ¡¡¡Sean amigos de SPICE!!! ¡¡¡Les conviene!!!
- Todos los programas de SPICE comparten:
  - un funcionamiento parecido
  - una sintaxis parecida
  - un conjunto similar de análisis posibles
- ¿Cuál es el mejor SPICE?
  - Depende del criterio
    - En términos de calidad /$, el mejor SPICE es el gratis
    - En términos de calidad absoluta, es discutible
  - El estándar de la industria microelectrónica es HSPICE
    - Originalmente comercializado por Meta Software (de Shawn y Kim Hailey), luego adquirida por Avant! Corporation, y ahora parte de Synopsys
    - HSPICE ha sido principalmente un programa de consola, sin interfaz gráfica

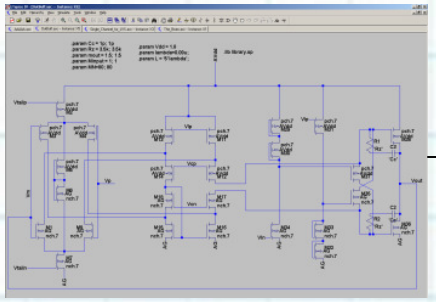
Breve historia de SPICE

- SPICE: Simulation Program with Integrated Circuit Emphasis
  - Desarrollado por Laurence Nagel y Donald Pederson en la Universidad de California, Berkeley
  - Presentado públicamente en 1973
  - De dominio público
- Basado en CANCER: Computer Analysis of Nonlinear Circuits, Excluding Radiation
  - Desarrollado también por Nagel en UC Berkeley
- SPICE fue concebido con fines docentes
  - Competía con programas concebidos con fines comerciales
  - Rápidamente alcanzó popularidad entre los estudiantes
  - Los estudiantes se graduaron, se fueron a la industria y naturalmente siguieron trabajando con SPICE
- Más información: *Life of SPICE*

http://www.designers-guide.org/perspective/life-of-spice.pdf

Entradas y salidas de SPICE

Capturador de esquemáticos



Deck de SPICE (texto), incluye netlist (listado de conexiones)

```
.param w=0.27u
.global AVdd
.lib library.sp
.ends spmos

.subckt inv_ckt In Out
Mp Out In $G_DVdd $G_DVdd pch.12 W='Wp', L
Mn Out In 0 0 nch.12 W='Wn', L='Lmin', m='
.lib library.sp
.param lambda = 0.09u
.param Wn = '6*lambda'
.param Wp = '10*lambda'
.param Mn = 'm_inv'
.param Mp = 'm_inv'
.param Lmin = '2*lambda'
.global DVdd
.global COM
.ends inv_ckt

.subckt 2nand_ckt In1 Out In2
Mp out In1 $G_DVdd $G_DVdd pch.12 W='Wp',
Mn P001 In1 0 0 nch.12 W='Wn', L='Lmin', m
Mp1 out In2 $G_DVdd $G_DVdd pch.12 W='Wp',
M1 out In2 P001 0 nch.12 W='Wn', L='Lmin',
.lib library.sp
.param lambda = 0.09u
.param Wn = '6*lambda'
.param Wp = '10*lambda'
.param Mn = 'm_2nand'
.param Mp = 'm_2nand'
.param Lmin = '2*lambda'
.global DVdd
.ends 2nand_ckt

.subckt extracted_csa_and_fb_subckt Vdd_CS
Cf_PhysCal1 Vout Vin 0.9p
Cf_NormOp1 N009 Vin 44.1p
XX1 Vout N009 N010 OpMode scmos params: n_
M1 Vout N013 Vin 0 nch.16 l=0.18u w=0.27u
M2 N009 N008 Vin 0 nch.16 l=0.18u w=0.36u
C1 N013 Vin 70f
C2 N012 Vin '17.6f'
XX3 AVdd N013 RST_Full_b RST_Full scmos pa
C3 N008 Vin 100f
C4 N004 Vin 17.6f
XX4 AVdd N008 RST_Full_b RST_Full scmos pa
XX10 N008 N004 N005 N006 scmos params: n_p
C5 N013 P001 30f
M3 N016 Vb6 AG AG nch.7 W=1.71u L=1.08u m=
M4 N015 Vbb Vdd_CSA Vdd_CSA pch.3 W=12.86u
M5 Vout Vb2 N015 Vdd_CSA pch.3 W=14.22u L=
M6 Vout Vnagicb N016 AG nch.7 w=1.71u l=1.
M7 N015 Vin AG AG nch.7 W=9.865u L=0.9u m=
XX12 N013 N012 N014 N007 scmos params: n_p
XX15 P001 Vin RST_Full RST_Full_b scmos pa
XX17 N012 Vin N011 N001 scmos params: n_pa
XX18 N004 Vin N002 N003 scmos params: n_pa
XX21 V_Ck Ph1_CSA Ph2_CSA ck_gen
XX14 N002 N003 inv_ckt_a
XX19 N001 N002 OpMode 2nand_ckt_a
XX11 OpMode N005 N007 2nand_ckt_a
XX20 N005 N006 inv_ckt_a
XX13 N007 N014 inv_ckt_a
XX22 OpMode N010 inv_ckt_a
XX2 N001 N011 inv_ckt_a
XX16 N007 N014 inv_ckt_a
XX6 Ph2_CSA P002 RST_Full_b 2nand_ckt_a
XX8 P002 N001 RST 2nand_ckt_a
XX7 Ph1_CSA P003 RST_Full_b 2nand_ckt_a
XX9 P003 N007 RST 2nand_ckt_a
.ends extracted_csa_and_fb_subckt

.subckt outbuff Vp Vout AVdd AG Vtail2 Vta
C2 Vout N013 'Cc'
C3 Vout N005 'Cc'
M2 N001 Vtail2 AVdd AVdd pch.7 l='L' w='36
M3 N014 Vout N001 AVdd pch.7 l='L' w='360*
M4 N012 Vp N001 AVdd pch.7 l='L' w='360*la
M5 N008 N008 N001 AVdd pch.7 l='L' w='360*
M9 N006 Vp N016 AG nch.7 l='L' w='NM*lambd
M11 N006 Vip AVdd AVdd pch.7 l='L' w='360*
M12 N002 Vcp N006 AVdd pch.7 l='L' w='360*
M13 N004 Vip AVdd AVdd pch.7 l='L' w='360*
M14 N009 Vcp N004 AVdd pch.7 l='L' w='360*
M15 N014 N009 AG AG nch.7 l='L' w='6A*lamb
M16 N012 N009 AG AG nch.7 l='L' w='6A*lamb
M17 N010 Vcn N012 AG nch.7 l='L' w='6A*lan
M18 N009 Vcn N014 AG nch.7 l='L' w='6A*lan
M29 N003 N003 AVdd AVdd pch.7 l='L' w='36
M30 N007 N007 N003 AVdd pch.7 l='L' w='360
M31 N011 Vip AVdd AVdd pch.7 l='L' w='360*
M32 N015 N015 AG AG nch.7 l='L' w='6A*lamb
M33 N011 N011 N015 AG nch.7 l='L' w='6A*la
M34 N007 Vin AG AG nch.7 l='L' w='6A*lambd
```

SPICE

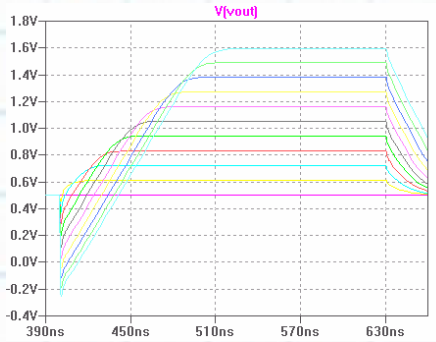
Resultados (texto)

```
Name:       m1
Model:      pch.2
Id:      -6.30e-05
Vgs:     -1.40e+00
Vds:     -1.00e+00
Vbs:      0.00e+00
Vth:     -4.41e-01
Vdsat:   -7.77e-01
Gm:       1.12e-04
Gds:      1.41e-06
Gmb       4.07e-05
Cbd:      8.44e-15
Cbs:      1.14e-14
Cgsov:    3.28e-15
Cgdov:    3.28e-15
Cgbov:    3.28e-16
dQgdVgb:  2.42e-13
dQgdVdb: -4.37e-15
dQgdVsb: -2.34e-13
dQddVgb: -5.46e-15
dQddVdb:  1.41e-14
dQddVsb:  3.01e-16
dQbdVgb: -3.12e-14
dQbdVdb: -8.92e-15
dQbdVsb: -4.65e-14
```

```
f0: v(inoise)=9.7611e-007 at 10
f1: v(inoise)=2.86965e-007 at 100
f2: v(inoise)=8.50399e-008 at 1000
f3: v(inoise)=2.73738e-008 at 10000
f4: v(inoise)=1.37805e-008 at 100000
f5: v(inoise)=1.19006e-008 at 1e+006
f6: v(inoise)=1.17242e-008 at 1e+007
f7: v(inoise)=1.17034e-008 at 1e+008
f8: v(inoise)=1.11925e-008 at 1e+009
f9: v(inoise)=3.62765e-009 at 1e+010
f10: v(inoise)=3.81484e-010 at 1e+011
f11: v(inoise)=3.95865e-011 at 1e+012
```

Out.raw

Visualizador de formas de onda



Análisis que SPICE puede realizar

- **Análisis DC**: cálculo no lineal del punto de operación
- **Análisis AC**: análisis lineal en el dominio de la frecuencia
- **Análisis transiente**: análisis no lineal en el dominio del tiempo
- **Análisis de curva de transferencia DC**: cálculo del punto de operación en función de un parámetro
- **Análisis de ruido**: análisis lineal en el dominio de la frecuencia
- **Análisis de función de transferencia**: cálculo de la ganancia de entrada a salida, e impedancias de un circuito
- **Análisis de Montecarlo**: cálculo de distribuciones estadísticas en alguna variable del circuito

Sintaxis de LTspice: lo básico (1/2)

- SPICE toma un “deck de SPICE” y produce resultados
- Un deck de SPICE define un listado de conexiones (netlist) y el análisis que queremos que SPICE ejecute
- Un deck de SPICE es un archivo de texto con extensión **`.sp`** o **`.net`**
- SPICE no distingue mayúsculas de minúsculas
- La primera línea de un deck de SPICE es siempre un comentario
- El deck de SPICE finaliza con **`.end`**, seguido de un salto de línea
- El orden de las líneas entre la inicial y **`.end`** es irrelevante
- Un netlist de SPICE especifica las componentes de un circuito, los nodos a los que va conectada cada componente, y sus parámetros
- Nodos y componentes son identificados por un nombre
  - El nombre de un nodo puede coincidir con el nombre de una componente
    - se diferencian según el contexto
- Cada línea en un deck de SPICE representa un comando o un elemento circuital. Ejemplos “en español”
  - Resistencia de nombre R1 entre los nodos V1 y V2, de 10kohm
  - Comando ejecutar simulación transiente con duración 10ms

Sintaxis de LTspice: lo básico (2/2)

- El primer carácter de una línea define el comando o elemento circuital especificado en la línea
  - Los “comandos punto” comienzan con `.`
  - Líneas con comentarios comienzan con **`*`**
  - Las líneas que comienzan con una letra instancian un elemento circuital
  - Las líneas que comienzan con `+` son la continuación de la línea anterior
- El carácter `;` significa que el resto de la línea es un comentario
- Respecto de los nombres de nodos y de elementos circuitales
  - Los nombres de los elementos circuitales comienzan con la letra que designa al tipo de elemento, seguida de caracteres alfanuméricos
    - Ejemplo: R1 o Cj2
  - Los nombres de nodos pueden ser combinaciones de caracteres alfanuméricos
    - En mediciones o visualización de formas de onda, el voltaje del nodo `x` es especificado como `V(x)`
- El nodo **`0`** es tierra global del circuito
- Los nombres de nodos que comienzan con **`$G_`** implican nodos globales

Sintaxis de LTspice: Tipos de línea según carácter inicial

*	Comentario
A	Dispositivo de función especial
B	Fuente definida por comportamiento
C	Capacitor
D	Diodo
E	Fuente de voltaje controlada por voltaje
F	Fuente de corriente controlada por corriente
G	Fuente de corriente controlada por voltaje
H	Fuente de voltaje controlada por corriente
I	Fuente de corriente independiente
J	Transistor JFET
K	Inductancia mutua
L	Inductor

M	Transistor MOSFET
O	Línea de transmisión con pérdidas
Q	Transistor bipolar
R	Resistor
S	Switch controlado por voltaje
T	Línea de transmisión sin pérdidas
U	Línea RC uniforme
V	Fuente de voltaje independiente
W	Switch controlado por corriente
X	**Subcircuito (instanciación)**
Z	Transistor MESFET
.	Directiva de simulación (comando punto)
+	Continuación de línea anterior

Sintaxis de LTspice: comandos punto

.AC	Análisis AC
.BACKANNO	Anotar nombre de nodo o corrientes de subcircuitos
.DC	Análisis DC
.END	Fin de netlist
.ENDS	Fin de definición de subcircuito
.FOUR	Cálculo de componente de Fourier
.FUNC	Funciones definidas por usuario
.FERRET	Descargar archivo dado un URL
.GLOBAL	Declaración de nodos globales
.IC	Definición de condiciones iniciales
.INCLUDE	Inclusión de otros archivos
.LIB	Inclusión de bibliotecas
.LOADBIAS	Cargar una solución DC previamente grabada
.MEASURE	Medición de cantidades eléctricas en simulación
.MODEL	Definición de un modelo de SPICE

.NET	Cálculo de parámetros de red en análisis AC
.NODESET	Definición de pistas para cálculo de condición inicial DC
.NOISE	Análisis de ruido
.OP	Análisis de punto de operación
.OPTIONS	Establece opciones de la simulación
.PARAM	Declaración de parámetros
.SAVE	Limitación de la cantidad de datos grabados
.SAVEBIAS	Grabación de punto de operación a archivo
.STEP	Repetición de simulación con diferentes parámetros
.SUBCKT	Definición de un subcircuito
.TEMP	Repetición de simulación con diferentes temperaturas
.TF	Cálculo de función de transferencia DC, pequeña señal
.TRAN	Análisis transiente
.WAVE	Grabación de resultados de nodos a archivo

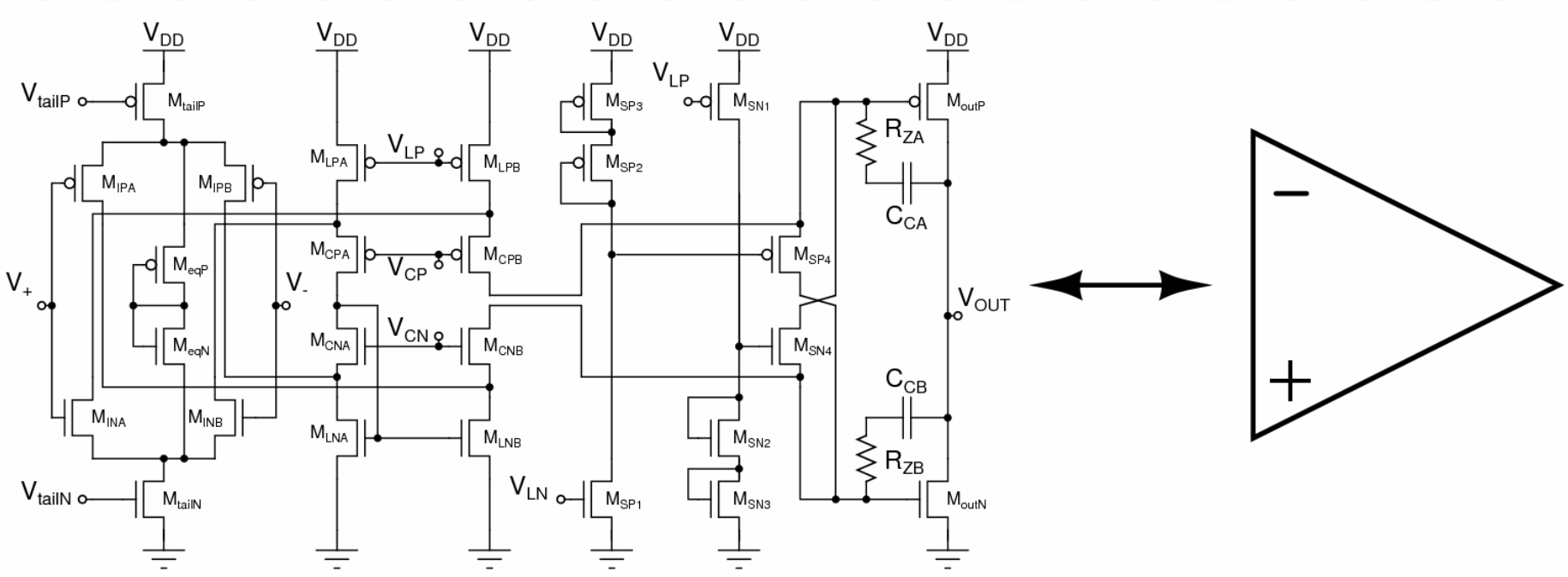
Sintaxis de LTspice: sufijos

Sufijo	Multiplicador
T	1e12
G	1e9
Meg	1e6
K	1e3
Mil	25.4e-6
m	1e-3
u	1e-6
n	1e-9
p	1e-12
f	1e-15

En realidad SPICE lee la primera letra del sufijo y descarta el resto, salvo con Mil o Meg

Modelos vs. Subcircuitos

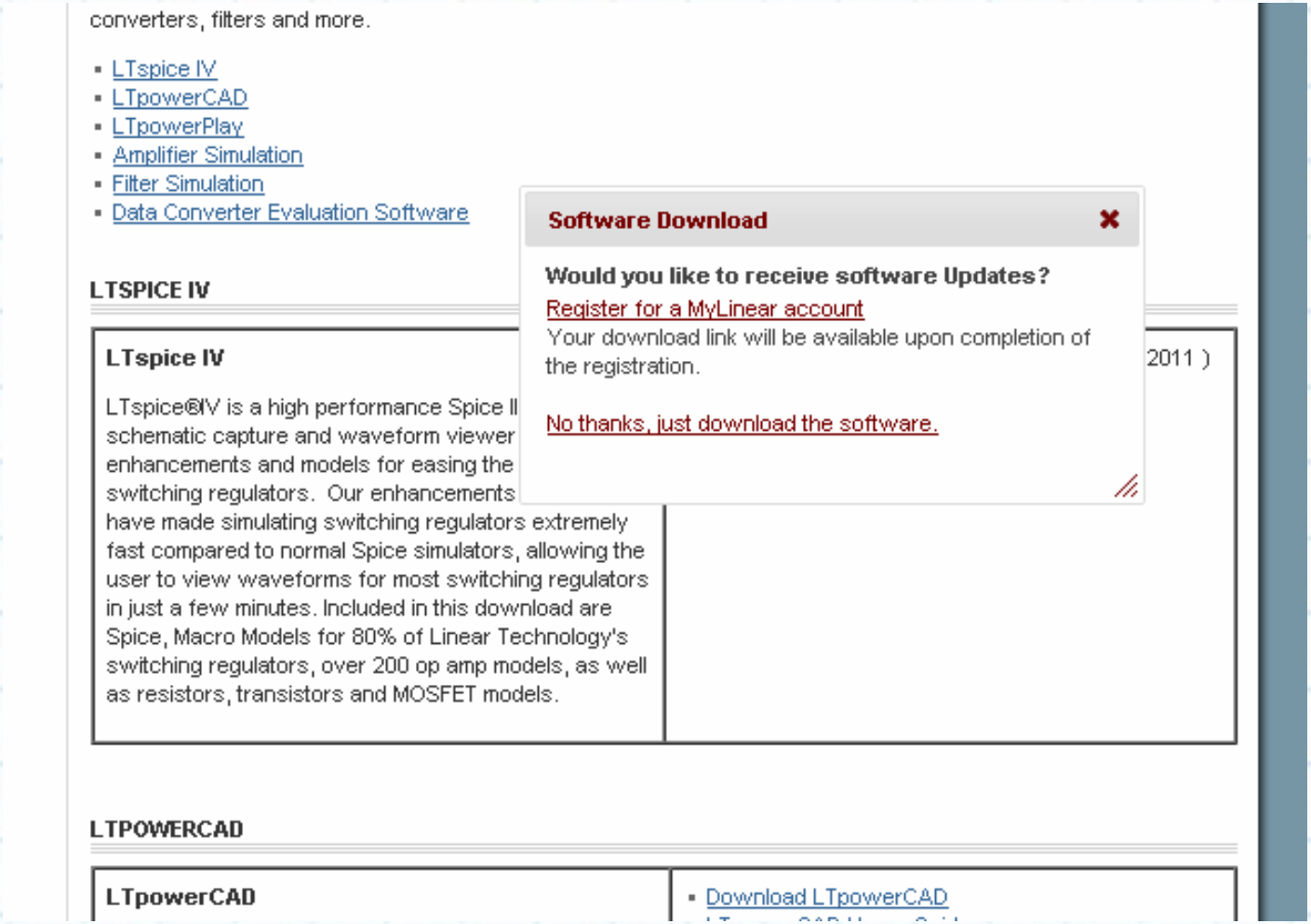
- Un modelo utiliza una de las primitivas de SPICE (ejemplo: transistor MOS, resistor, diodo, etc.), y su nivel de complejidad depende de los parámetros definidos
- Los subcircuitos son ideales para trabajar con circuitos jerárquicos
  - Más adelante veremos de qué se trata…
- Un subcircuito es un circuito que puede utilizar cuaquier elemento de SPICE, y puede tener un nivel de complejidad arbitrario
  - Ejemplo: el subcircuito de un amplificador operacional



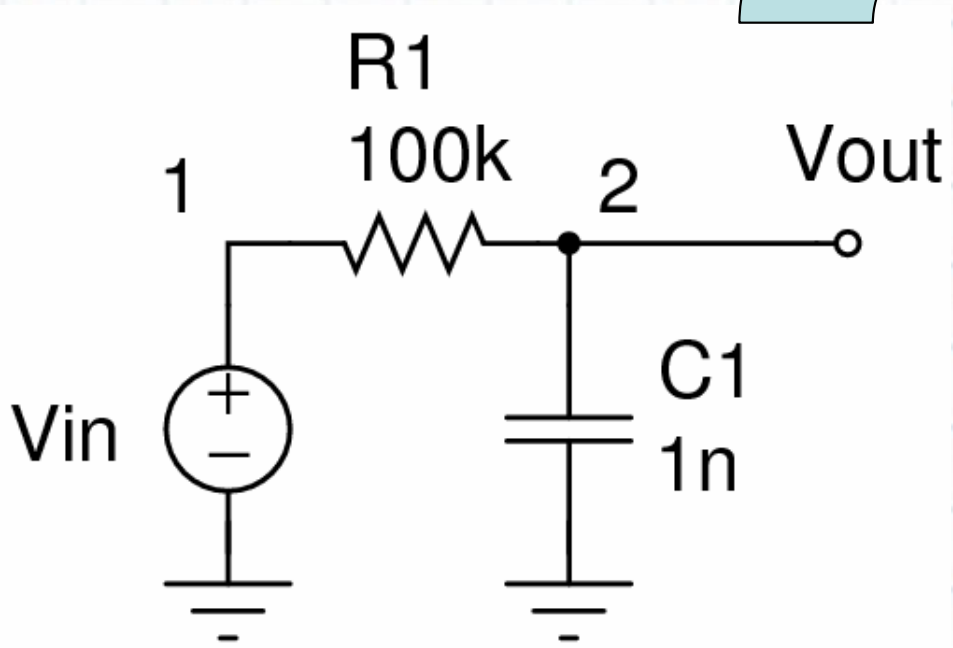
Primeros pasos con LTspice

Descarga, instalación y ejecución

http://ltspice.linear.com/software/LTspiceIV.exe



Un listado de conexiones muy simple

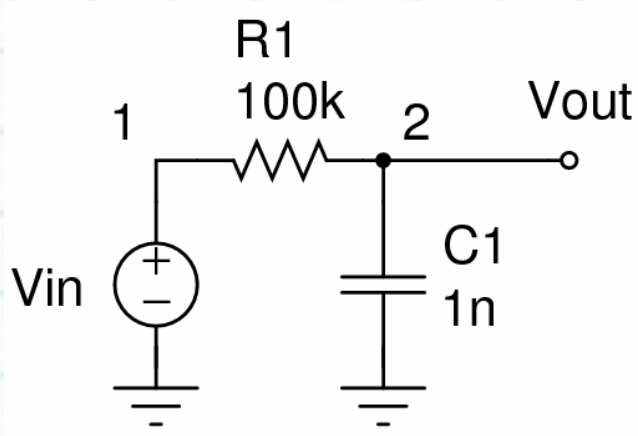



```
.param Vin = 1V
Vin 1 0 'Vin'
R1 2 1 100k
C1 2 0 1n
```

- Uno puede crear el listado de conexiones en cualquier editor de texto (notepad)
- Si ya existe el listado de conexiones, es posible editarlo directamente en LTspice

Ej. 1: Análisis Pto. de operación

```
Analisis Pto Op.
.param Vin = 1V
Vin 1 0 'Vin'
R1 2 1 100k
C1 2 0 1n
.op
.end
```



Resultado

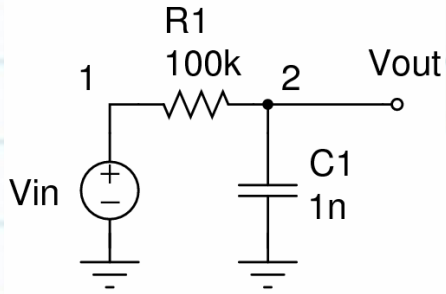
```
 --- Operating Point ---

V(1):                1                      voltage
V(2):                1                      voltage
I(C1):  1e-021               device_current
I(R1):  -2.22045e-021        device_current
I(Vin):          -1.69407e-021  device_current
```

Procedimiento

- Ejecutamos la simulación

Ej. 2: Análisis transiente

```
Analisis Transiente

.param Vin = 1V

Vin 1 0  pwl ( 0        0
+              1m       0
+              1.001m   'Vin'
+              2m       'Vin'
+              2.001m   0
+              3m       0)

R1 2 1 100k
C1 2 0 1n

.tran 4m

.end
```

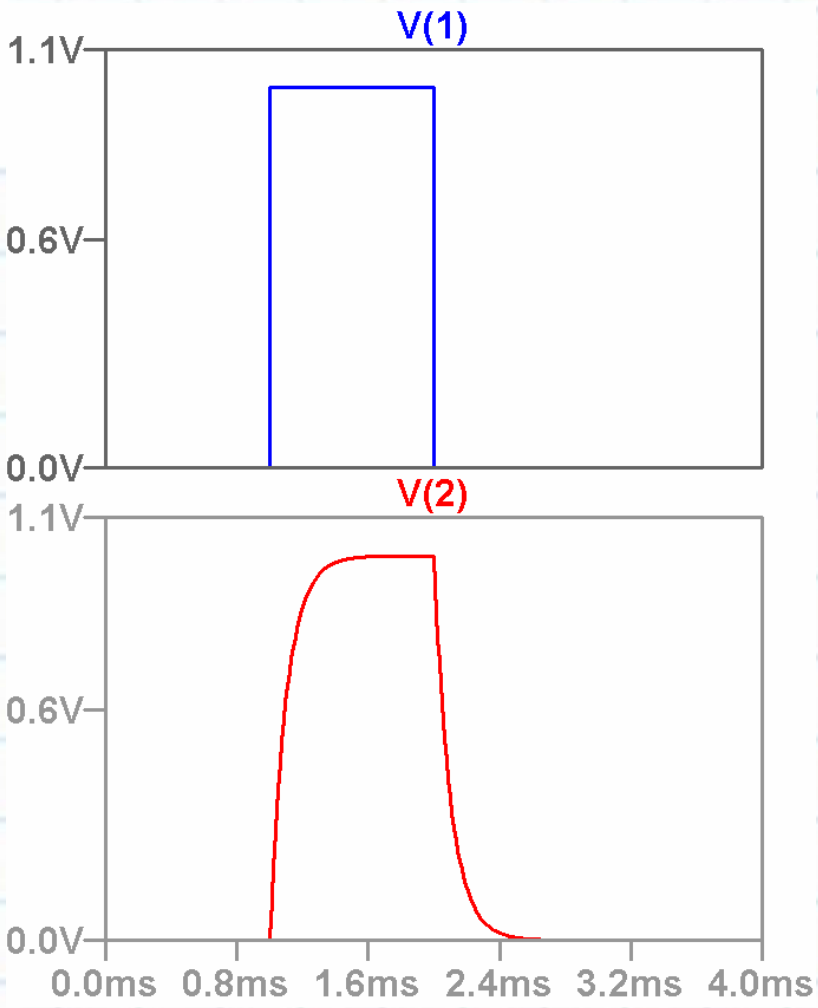


Procedimiento

Ejecutamos la simulación

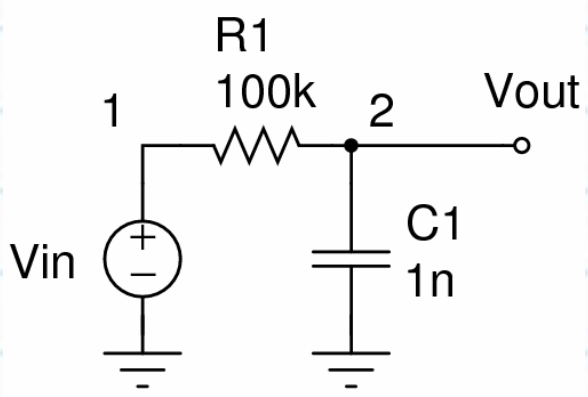
Elegimos voltajes o corrientes para visualizar

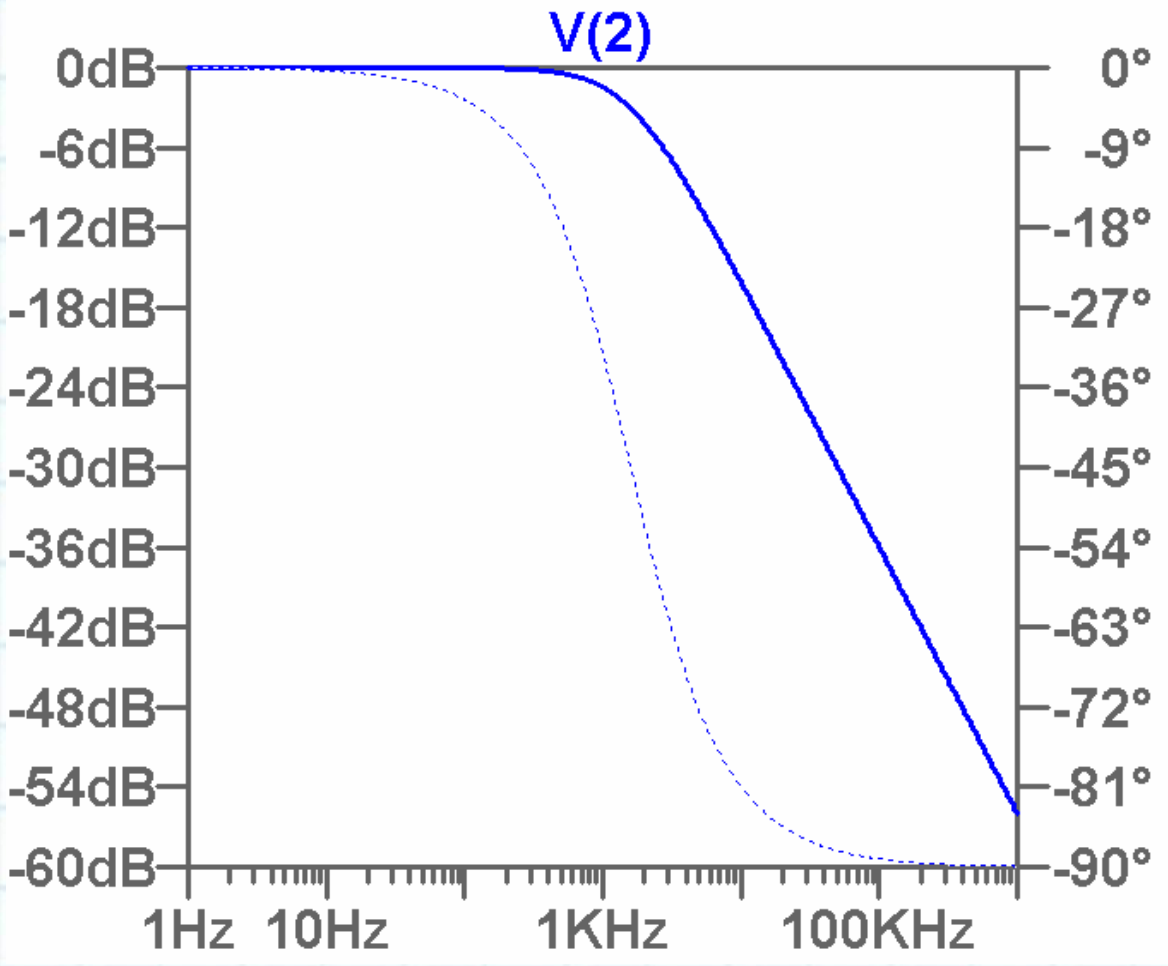
Resultado



Ej. 3: Análisis AC

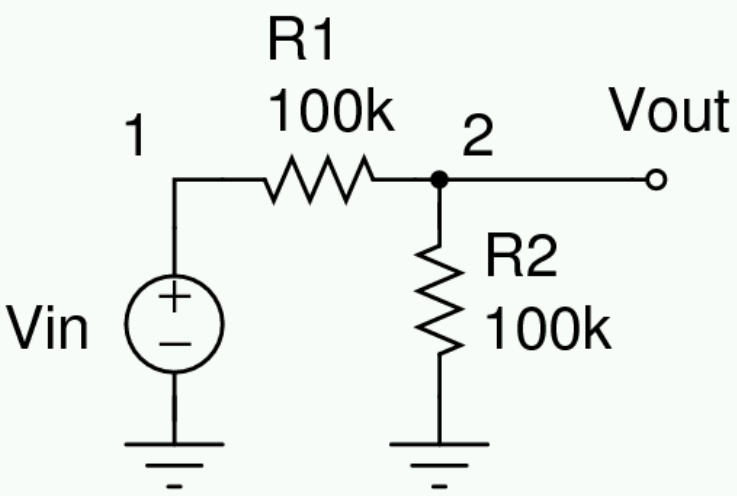
```
Analisis AC

Vin 1 0 AC 1
R1 2 1 100k
C1 2 0 1n

.ac dec 100 1 1meg

.end
```

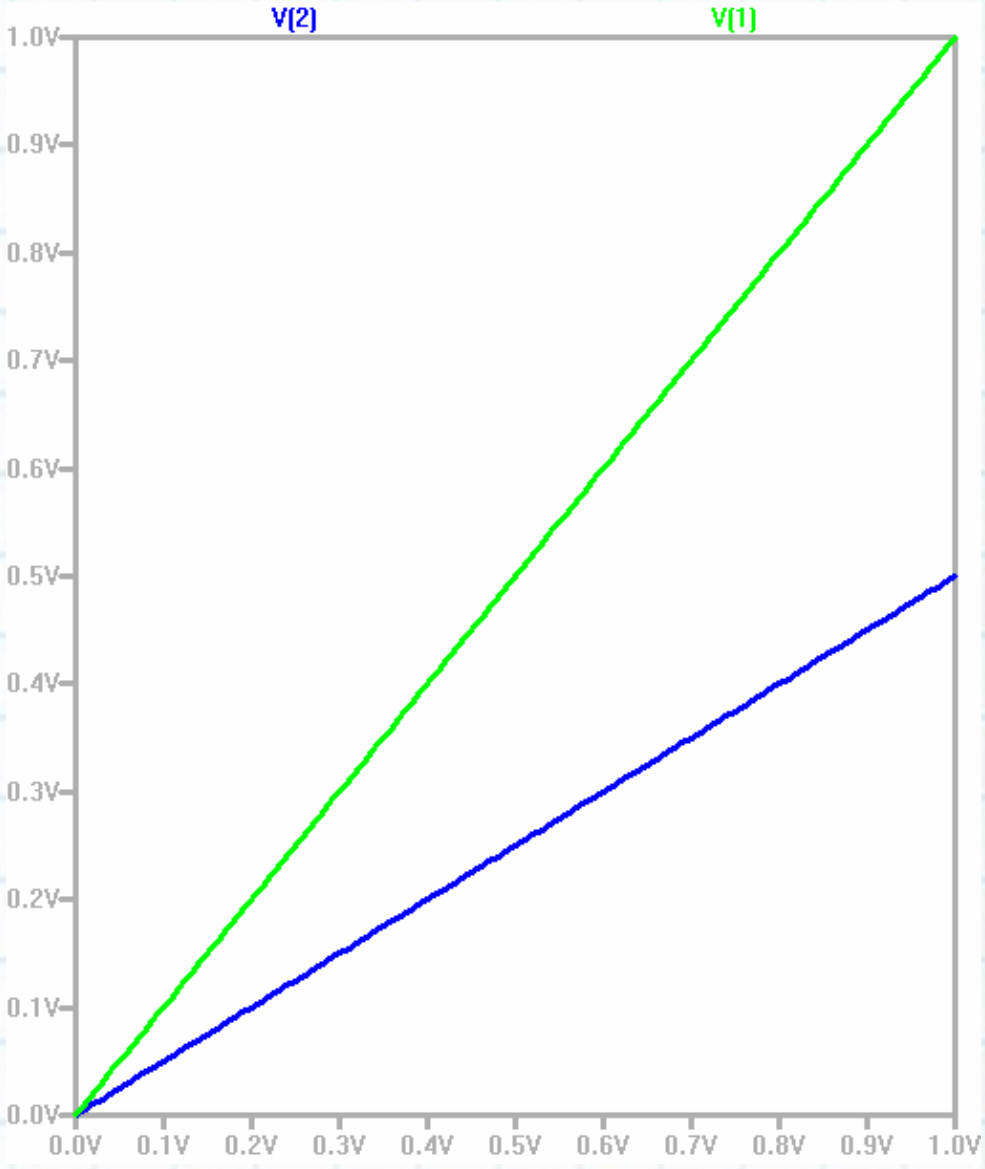




Ej. 4: Análisis de curva de transferencia DC

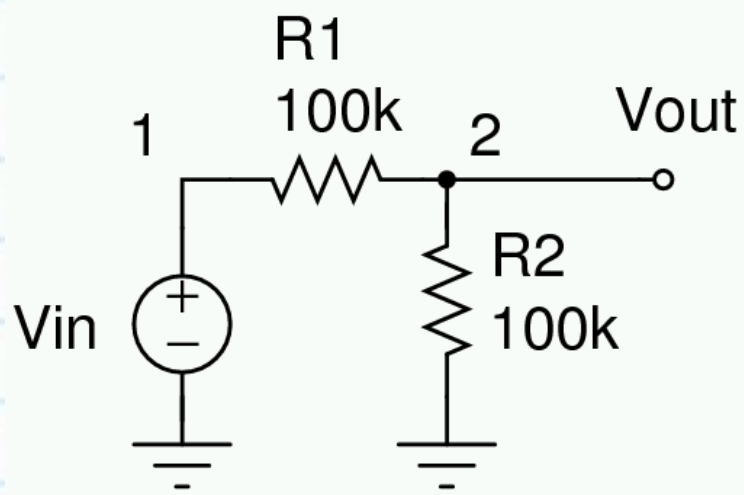
```
Analisis DC

.param Vin = 1V
Vin 1 0 'Vin'
R1 2 1 100k
R2 2 0 100k

.dc Vin 0 1 0.01

.end
```





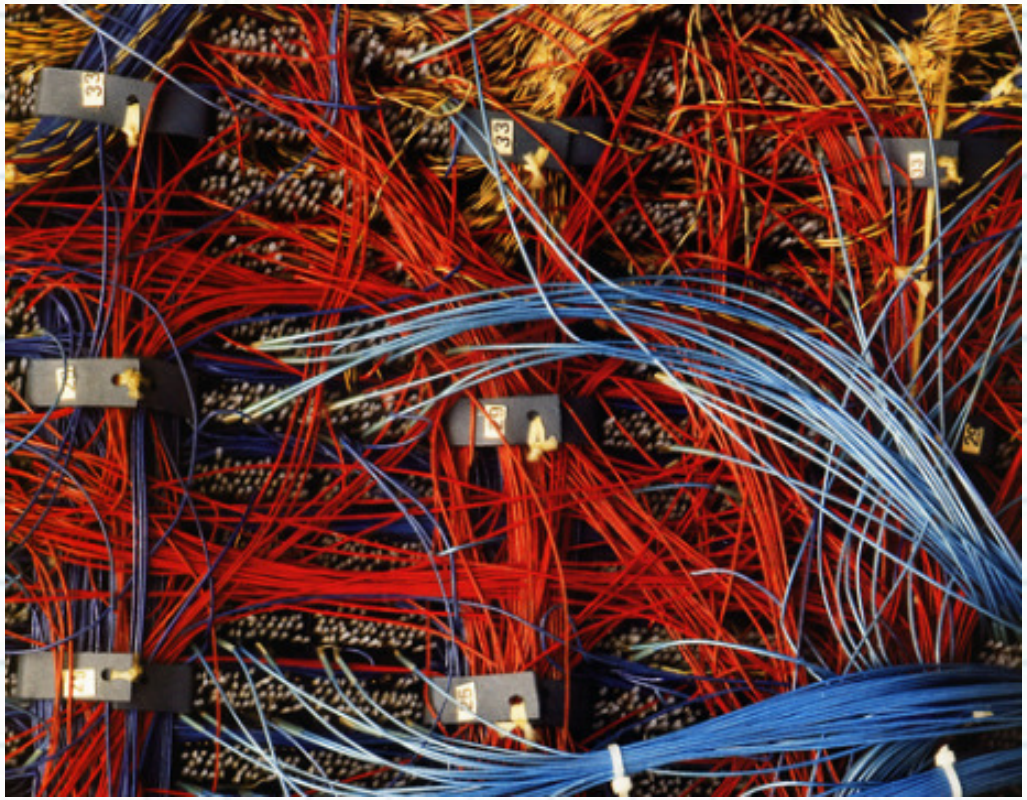
Ej. 5: Análisis de función de transferencia: DC (frecuencia 0), pequeña señal

```
Analisis Fcn. Transf.
.param Vin = 1V
Vin 1 0 'Vin'
R1 2 1 100k
R2 2 0 100k
.tf V(2) Vin
.end
```




```
 --- Transfer Function ---

Transfer_function:           0.5          transfer
vin#Input_impedance:         200000       impedance
output_impedance_at_V(2):    50000        impedance
```

Hilando un poco más fino



Capturador de esquemáticos

- En general, es tedioso trabajar con circuitos de más de un par de nodos, usando solo texto
  - Es fácil confundirse y equivocarse
  - No es intuitivo
  - No es práctico para compartir un circuito…
- LTspice convenientemente trae un capturador de esquemáticos
  - Muy intuitivo y fácil de utilizar
  - Permite utilizar componentes que no vienen con LTspice
- Internamente, LTspice genera un netlist a partir del esquemático cada vez que ejecutamos una simulación
  - Podemos ver ese netlist mediante el comando View → SPICE Netlist
- ¿Y de qué me sirve entender un netlist, si LTspice ya incluye un capturador de esquemáticos muy bueno?
  - Entender un netlist es fundamental para trabajar seriamente con SPICE
    - Muchos comandos, comentarios y parte del código del netlist aparece en el esquemático
    - Al final, la entrada que uno entrega a SPICE siempre se reduce a un netlist, ¡incluso en circuitos de miles de transistores!
    - Y muchas veces no tenemos acceso a un esquemático, como en el caso de un circuito extraído a partir de un layout

Algunos Shortcuts de LTspice (personalizables)

Capturador de esquemáticos

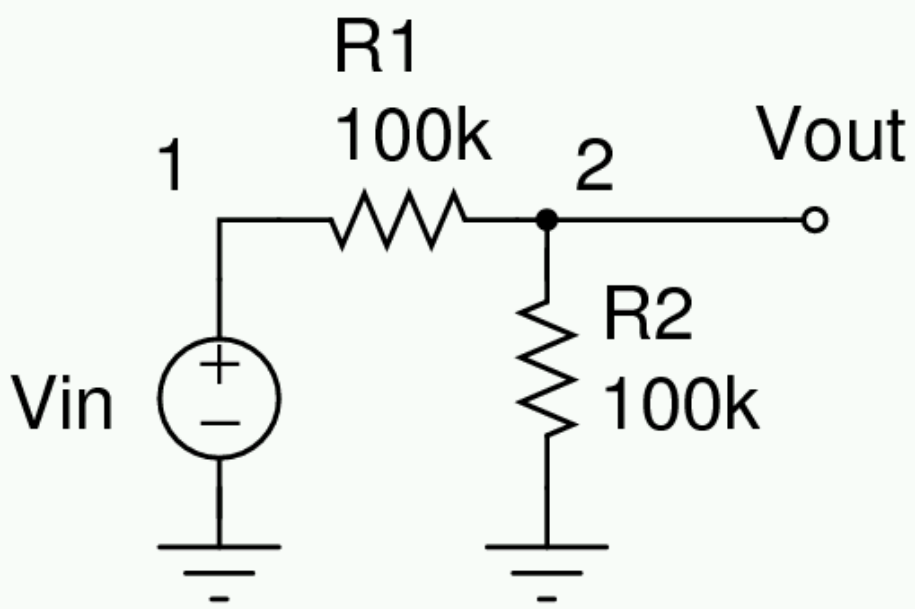
F1	Ayuda (excelente)
F2	Insertar componente
F3	Dibujar conexión
F4	Insertar etiqueta
F5	Borrar
F6	Copiar
F7	Mover (desconectando)
F8	Mover (sin desconectar)
F9	Deshacer
Shift+F9	Rehacer
T	Insertar comentario de texto
S	Insertar directiva de SPICE
G	Insertar nodo de tierra
R	Insertar resistor

C	Insertar capacitor
L	Insertar inductor
D	Insertar diodo
Ctrl+R	Rotar componente seleccionada
Ctrl-E	Invertir componente seleccionada

Visualizador de formas de onda

Ctrl-Y	Autorango vertical
Ctrl-Z	Zoom área
F5	Borrar traza
Ctrl-A	Agregar traza
F9	Deshacer
Shift+F9	Rehacer
Ctrl+H	Detener simulación

Veamos un ejemplo…



Netlist plano vs. Netlist jerárquico; Subcircuitos

- Netlist plano: no incluye subcircuitos
  - Cambios en una instancia de un circuito no afecta otras instancias del mismo circuito
- Netlist jerárquico: incluye subcircuitos
  - Cada vez que un subcircuito es cambiado, todas las instancias de ese subcircuito sufren el mismo cambio
- Ejemplo de definición de subcircuitos

```
.subckt Mi_Inversor in out num=Wn razon=2
Mn1 out in 0 0 NMOS w='num' L='Lmin'
+ PD = 'num+6*Lmin', PS = '2*num+Lmin'
+ AD = '3*num*Lmin', AS = '3*num*Lmin'
Mp1 out in 0 0 PMOS w='num' L='Lmin'
+ PD = 'num*razon+6*Lmin', PS = '2*num*razon+Lmin'
+ AD = '3*num*razon*Lmin', AS = '3*num*razon*Lmin'
.ends
```

- Ejemplo de instanciación de subcircuito

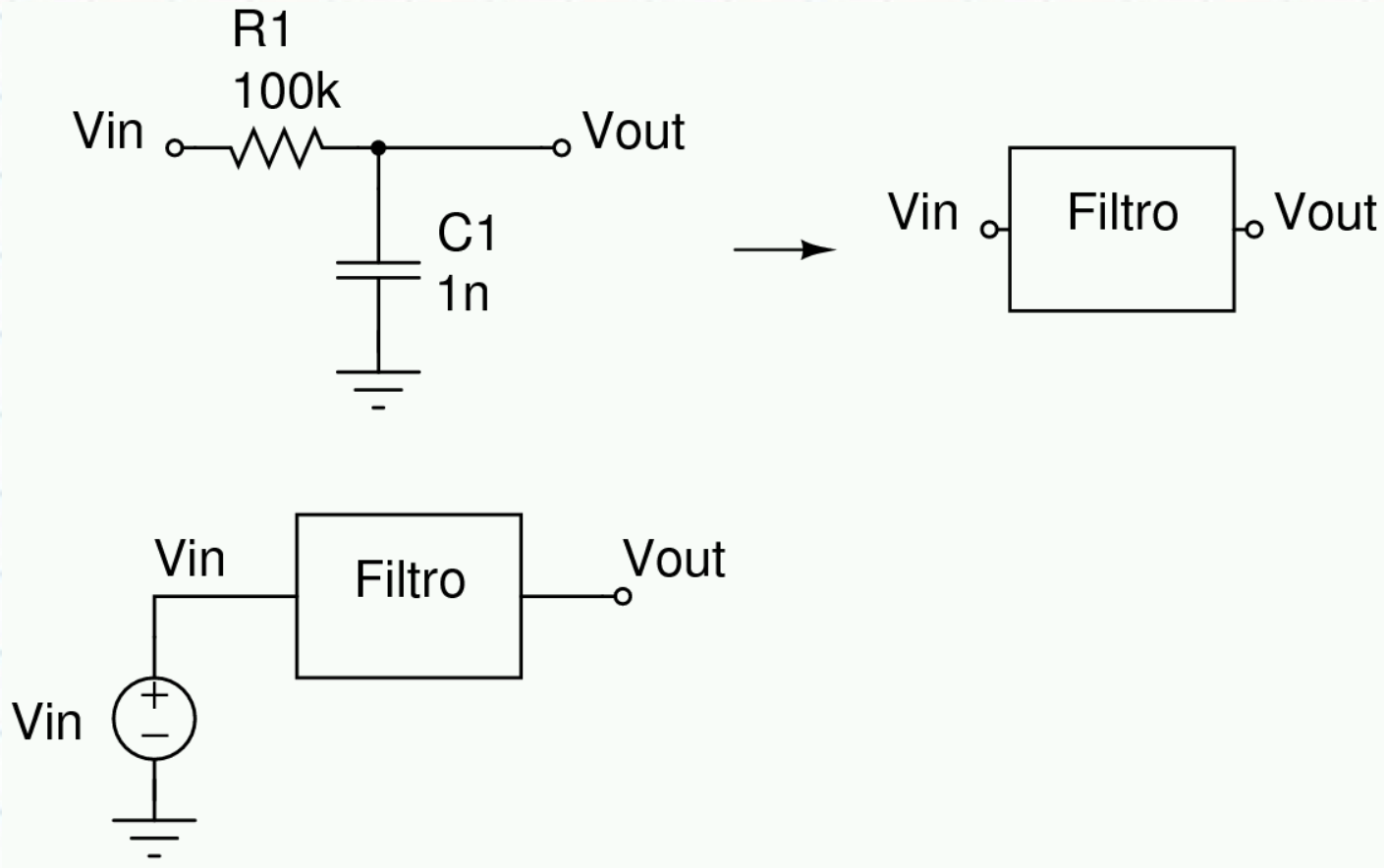
```
Xinv1 in out Mi_Inversor num=Wn razon=3
```

Jerarquía en el capturador de esquemáticos

- El capturador de esquemáticos de LTspice maneja jerarquías de manera natural
- Doble click sobre un símbolo abre el subcircuito correspondiente
- Es posible crear símbolos o dejar que LTspice cree un símbolo para un esquemático
  - en ese caso, hay que etiquetar los nodos que corresponden a entradas y salidas usando F4
- El subcircuito y su símbolo asociado tienen el mismo nombre, con extensión `.asc` y `.asy`, respectivamente
- Es posible definir atributos sobre un subcircuito y crear una ventana de diálogo que permita editar sus parámetros
- Es posible usar un símbolo existente y asociarlo a un subcircuito proporcionado por un fabricante
  - Ejemplo: es posible bajar de la web subcircuitos de “macromodelos” de chips, tales como amplificadores operacionales
  - Luego podemos simular un circuito que incluya el amplificador exacto a utilizar
  - Existen miles de macromodelos para los chips más utilizados en el mundo

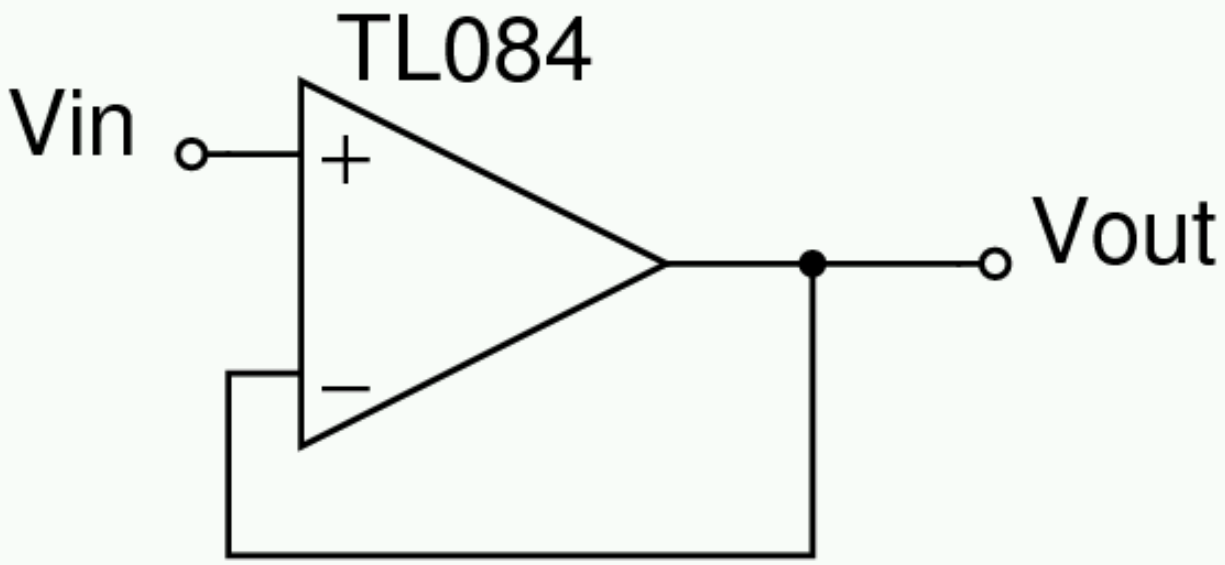
Ej. 6: Subcircuitos

- Crearemos un subcircuito “filtro”
- Luego utilizaremos el filtro en una simulación



Ej. 7: Modelos proporcionados por el fabricante

- Descargaremos un macromodelo del amplificador operacional TL084 de la página del fabricante
- Luego usaremos ese modelo en nuestro circuito y haremos algunos experimentos



Mediciones (`.meas`)

- SPICE permite realizar mediciones sobre la simulación ya realizada
- La medición puede ser especificada
  - directamente en el deck de SPICE, o
  - en un archivo aparte
- Se utiliza el comando `.meas`
- Ejemplos de posibles mediciones:
  - Encontrar el voltaje de un nodo en un instante de tiempo determinado
  - Encontrar el valor de una expresión en función de voltajes y corrientes del circuito, la primera vez que otro voltaje del circuito cruce un cierto umbral
  - Encontrar el valor de una expresión en función de voltajes y corrientes del circuito, la última vez que otro voltaje del circuito cruce un cierto umbral en su flanco de subida
  - Encontrar el valor de una expresión en función de voltajes y corrientes del circuito, la tercera vez que otro voltaje del circuito cruce un cierto umbral, contando desde un punto arbitrario en el tiempo
  - Encontrar una expresión aritmética en función de otras mediciones
  - Encontrar el momento en que un voltaje cruza cierto umbral…
  - Encontrar el ancho de un pulso
  - Encontrar el valor pico de una forma de onda
  - Etc.
- Se recomienda explorar las diferentes posibilidades de esta herramienta, que está bien documentada en el manual de LTspice

Otras opciones de SPICE…

- Aritmética, disponible para
  - Formas de onda
  - Definición de parámetros
  - Mediciones
- Uso de `.wav` como estímulo, o grabación de un `.wav`
- Convergencia y punto de operación
  - `.ic` fija condiciones iniciales
  - `.savebias` graba punto de operación
  - `.loadbias` carga un punto de operación
    - Usar `.ic` en lugar de `.nodeset`
- Inclusión de archivos
  - Bibliotecas: usar `.lib`
  - Otros archivos: usar `.include`
- Operación por línea de comandos
  - `C:\Program Files\LTC\LTspiceIV>` **`scad3 -b archivo.sp`**

Más información

¿Dónde puedo conseguir más información?

- Manual de LTspice
- Manual de HSPICE
- Grupo de Yahoo de LTspice
- Laurence Nagel: *The Life of SPICE*

http://www.designers-guide.org/perspective/life-of-spice.pdf