SPICE

Pravin Zode

Outline

- Introduction to Design
- Types of Analysis
- Scale factors
- Element Conventions
- Circuit Descriptions
- Examples

Introduction

- SPICE (Simulation Program with Integrated Circuit Emphasis)
- It is powerful circuit simulation tool that allows engineers to analyze and design electronic circuits with great accuracy
- SPICE allows for the creation of accurate circuit models to simulate real-world behavior
- Accurate prediction of circuit behavior without physical prototypes
- SPICE can perform a variety of analyses, including DC, AC, transient, and more
- Saves time and cost by detecting issues early in the design process.

Types of Analysis

- **DC Analysis** : SPICE can determine the steady-state behavior of analog circuits, such as bias points and power consumption
- **AC Analysis** : SPICE can simulate the frequency response of analog circuits, including gain, phase, and bandwidth
- **Transient Analysis** : SPICE can model the dynamic behavior of analog circuits, including response to step inputs and oscillations

Digital Circuit Simulation

- SPICE can accurately simulate the behavior of digital logic gates, including propagation delays and logic levels
- SPICE can model the behavior of flip-flops, registers, and other sequential logic circuits
- SPICE can perform timing analysis on digital circuits, ensuring proper operation and identifying potential timing issues

Advanced Simulation Features

- **Monte Carlo Analysis** : SPICE can perform statistical analysis to assess the impact of component variations on circuit performance
- **Parametric Sweeps** : SPICE allows for the simulation of circuits with varying parameter values, enabling design optimization
- **Optimization Algorithms** : SPICE can be coupled with optimization algorithms to automate the design process

Scale Factors

Suffix	Name	Factor
T	Tera	10^{12}
G	Giga	10^{9}
Meg	Mega	10^{6}
K	Kilo	10^{3}
mil	Mil	25.4×10^{-6}
m	milli	10^{-3}
u	micro	10^{-6}
n	nano	10^{-9}
p	pico	10^{-12}
f	femto	10^{-15}
a	atto	10^{-18}

Element Conventions

A	XSPICE code model
B	Behavioral (arbitrary) source
C	Capacitor
D	Diode
E	Voltage-controlled voltage source (VCVS)
F	Current-controlled current source (CCCs)
G	Voltage-controlled current source (VCCS)
H	Current-controlled voltage source (CCVS)
I	Current source
J	Junction field effect transistor (JFET)
K	Coupled (Mutual) Inductors
L	Inductor
M	Metal oxide field effect transistor (MOSFET)

N	Numerical device for GSS
O	Lossy transmission line
P	Coupled multiconductor line (CPL)
Q	Bipolar junction transistor (BJT)
R	Resistor
S	Switch (voltage-controlled)
T	Lossless transmission line
U	Uniformly distributed RC line
U	Basic digital building blocks using XSPICE
V	Voltage source
W	Switch (current-controlled)
X	Subcircuit
Y	Single lossy transmission line (TXL)
Z	Metal semiconductor field effect transistor (MESFET)

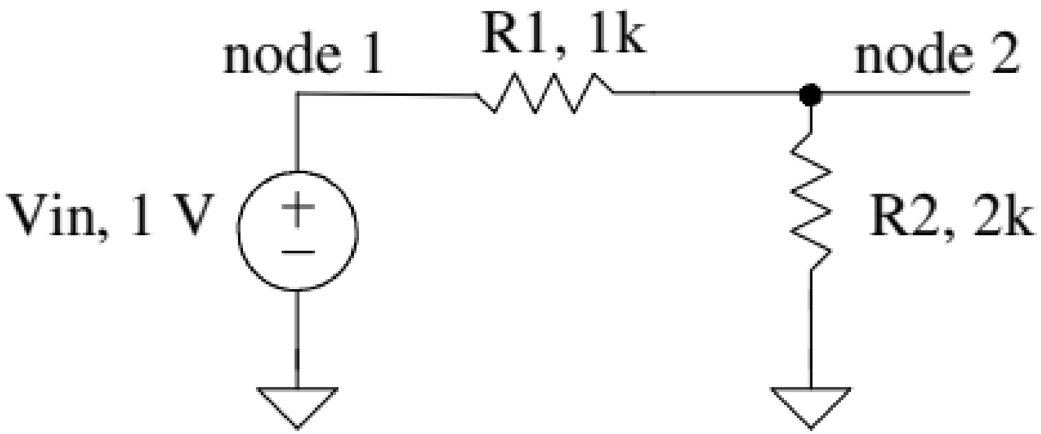
Circuit Description

- The first line in the input file must be the title, which is the only comment line that does not need any special character in the first place
- The last line must be .end, plus a newline delimiter
- Commands are specified by Dot Command (.dc, .ac etc)
- Node "0" is always the ground (reference) node.
- Components are identified by letters based on their type, followed by numbers (e.g., R1, C1, M1)
- Comments are declared followed by ;
- A line may be continued by entering a ‘+’ (plus) in column 1 of the following line

Structure of SPICE Netlist

- Title
- Controls
- Sources
- Model
- Components
- Subcircuit

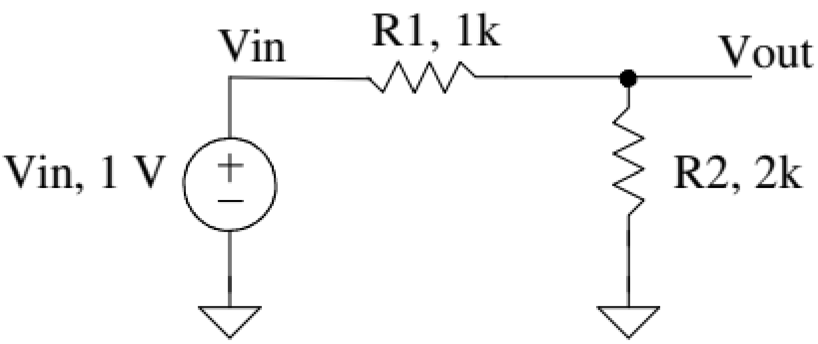
Example-01 (Netlist)




```
Vin     1     0     DC     1
R1      1     2     1k
R2      2     0     2k

.end
```

```
v(1) = 1.000000e+00
v(2) = 6.666667e-01
vin#branch = -3.33333e-04
```

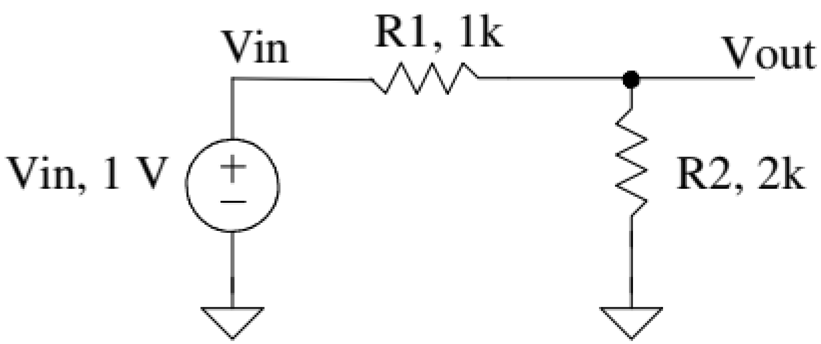
Example-02 (Node Name , op)




```
*#destroy all
*#run
*#print all
.op
Vin Vin 0 DC 1
R1 Vin Vout 1k
R2 Vout 0 2k
.end
```

```
v(1) = 1.000000e+00
v(2) = 6.666667e-01
vin#branch = -3.33333e-04
```

Example-03 (Transfer Function)



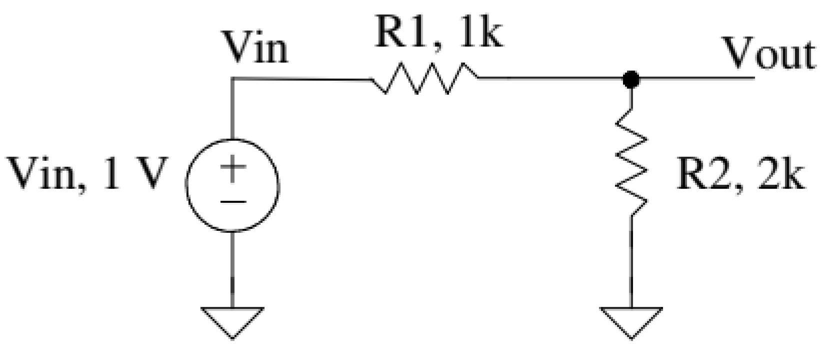

```
*#destroy all
*#run
*#print all
.op
Vin Vin 0 DC 1
R1 Vin Vout 1k
R2 Vout 0 2k
.end
```

```
.TF      V(Vout,0)      Vin
```

```
transfer_function = 6.666667e-01
output_impedance_at_v(vout,0) = 6.666667e+02
vin#input_impedance = 3.000000e+03
```

- "gain" of this voltage divider is 2/3
- Input resistance is 3k = (1k + 2k)
- Output resistance is 667= (1k||2k)

Example-04 (Transfer Function)



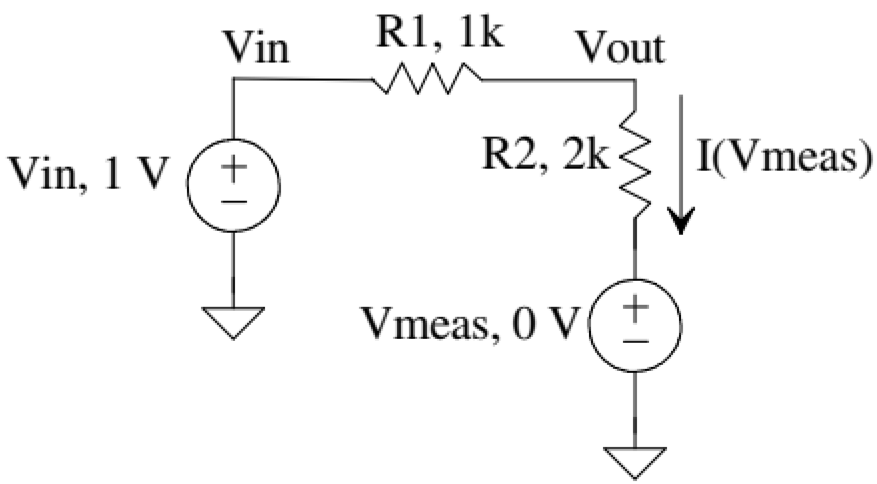

```
*#destroy all
*#run
*#print all
.op
Vin Vin 0 DC 1
R1 Vin Vout 1k
R2 Vout 0 2k
.end
```

```
.TF      V(Vout,0)      Vin
```

```
transfer_function = 6.666667e-01
output_impedance_at_v(vout,0) = 6.666667e+02
vin#input_impedance = 3.000000e+03
```

- "gain" of this voltage divider is 2/3
- Input resistance is 3k = (1k + 2k)
- Output resistance is 667= (1k||2k)

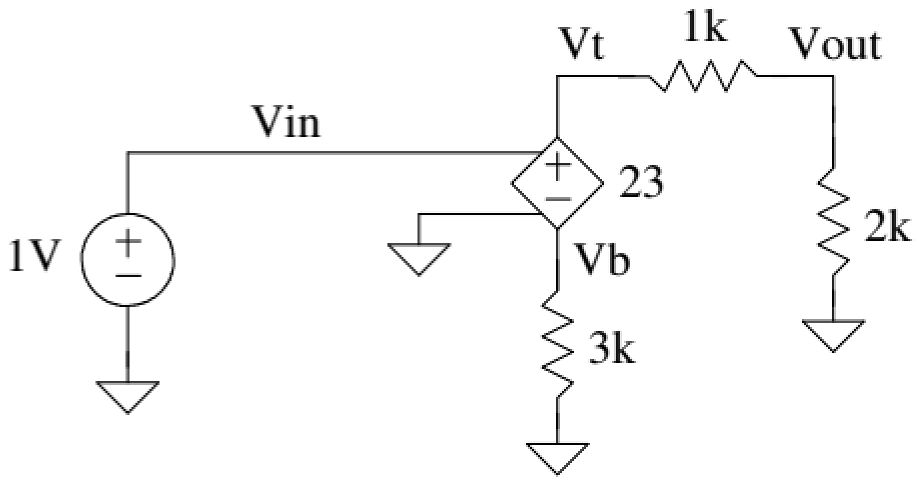
Example-04 (Transfer Function)




```
*#destroy all
*#run
*#print all
.TF I(Vmeas) Vin
Vin Vin 0 DC 1
R1 Vin Vout 1k
R2 Vout Vmeas 2k
Vmeas Vmeas 0 DC 0
.end
```

The gain is I(Vmeas)/Vin or 1/3k (= 333 µmhos)

Example-05 (VCVS)

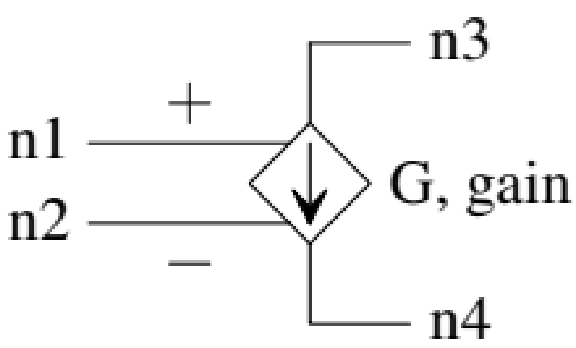



```
.TF     V(Vout,0) Vin

Vin     Vin     0       DC      1
R1      Vb      0       3k
R2      Vt      Vout    1k
R3      Vout    0       2k
E1      Vt      Vb      Vin     0       23

.end
```

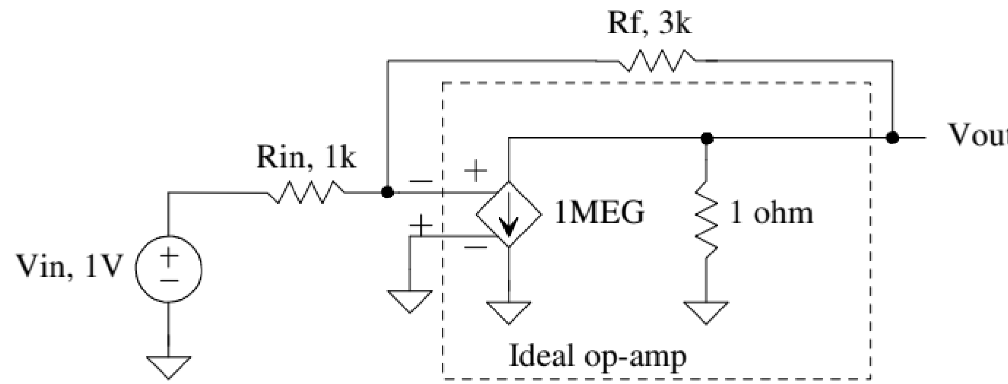
```
transfer_function = 7.666667e+00
output_impedance_at_v(vout,0) = 1.333333e+03
vin#input_impedance = 1.000000e+20
```

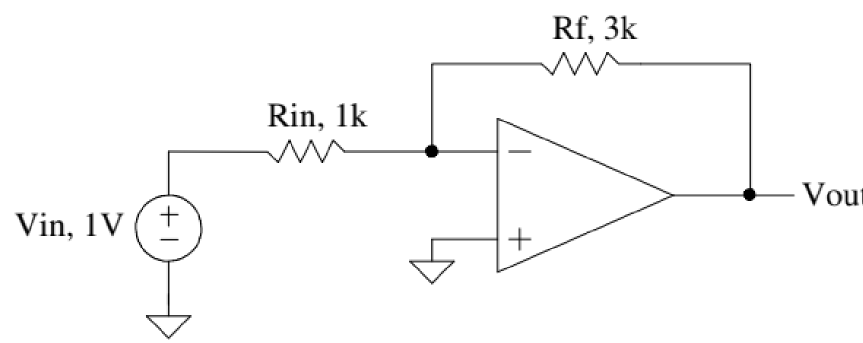
Example-06 (Ideal OpAmp)

- Ideal op-amp can be implemented in SPICE with a VCVS or with a voltage controlled current source (VCCS)

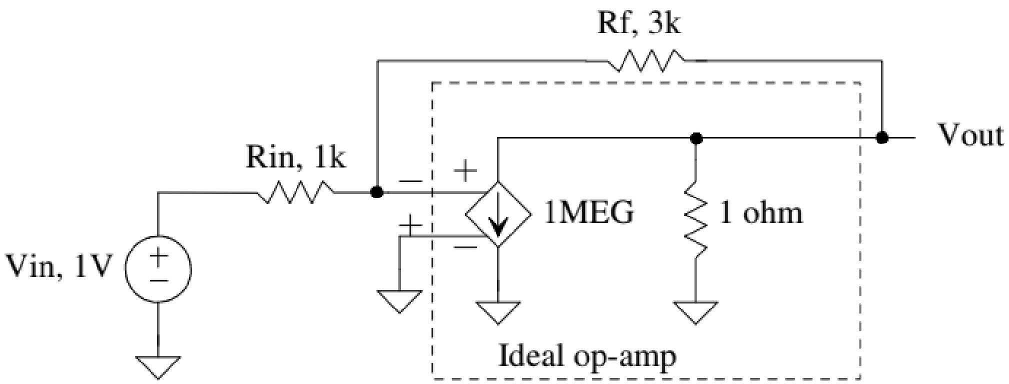


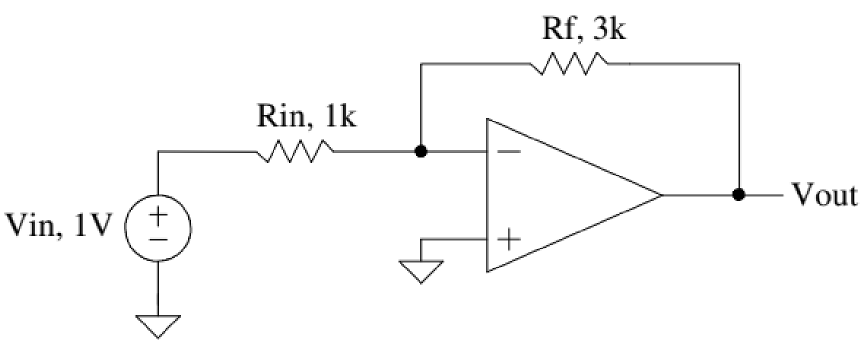
Voltage-Controlled Current Source (VCCS)
G1 n3 n4 n1 n2 G





Example-07 (Sub Circuit)

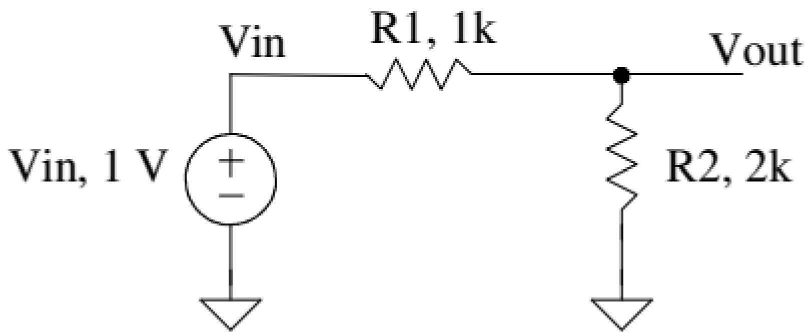




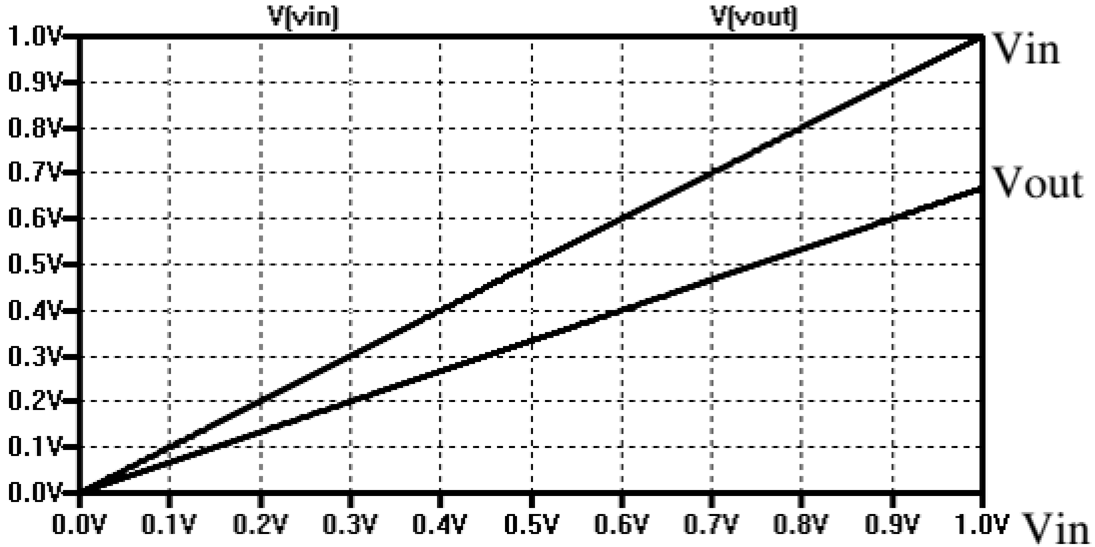

```
*#destroy all
*#run
*#print all

.TF       V(Vout,0) Vin

Vin       Vin       0         DC        1
Rin       Vin       Vm        1k
Rf        Vout      Vm        3k

X1        Vout      0         vm        Ideal_op_amp

.subckt Ideal_op_amp  Vout    Vp        Vm
G1        Vout      0         Vm        Vp        1MEG
RL        Vout      0         1
.ends
.end
```

```
transfer_function = -2.99999e+00
output_impedance_at_v(vout,0) = 3.999984e-06
vin#input_impedance = 1.000003e+03
```

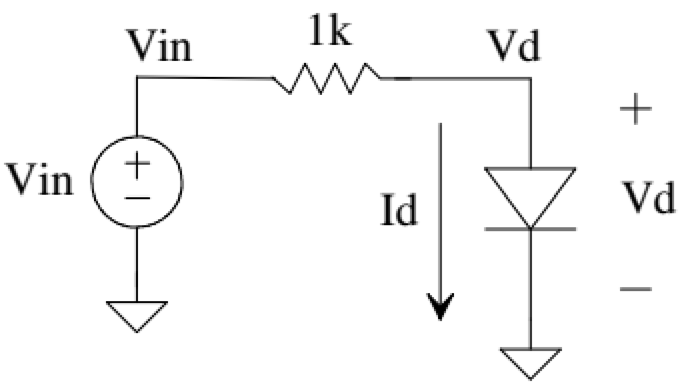
Example-08 (DC Analysis)



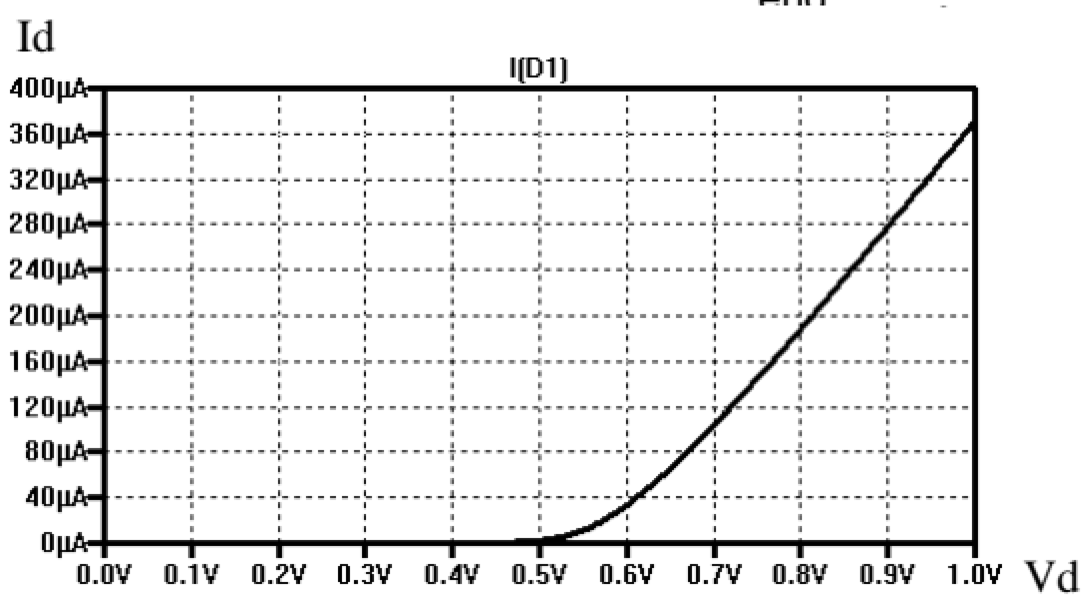

```
*#destroy all
*#run
*#plot Vin Vout
.dc Vin 0 1 1m
Vin Vin 0 DC 1
R1 Vin Vout 1k
R2 Vout 0 2k
.end
```



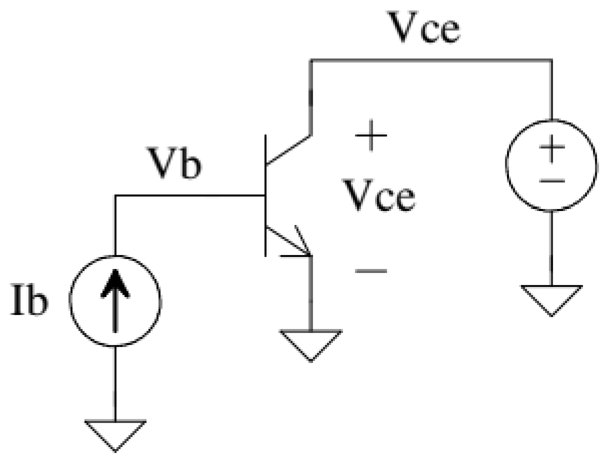
Example-09 (Plotting IV Curve)



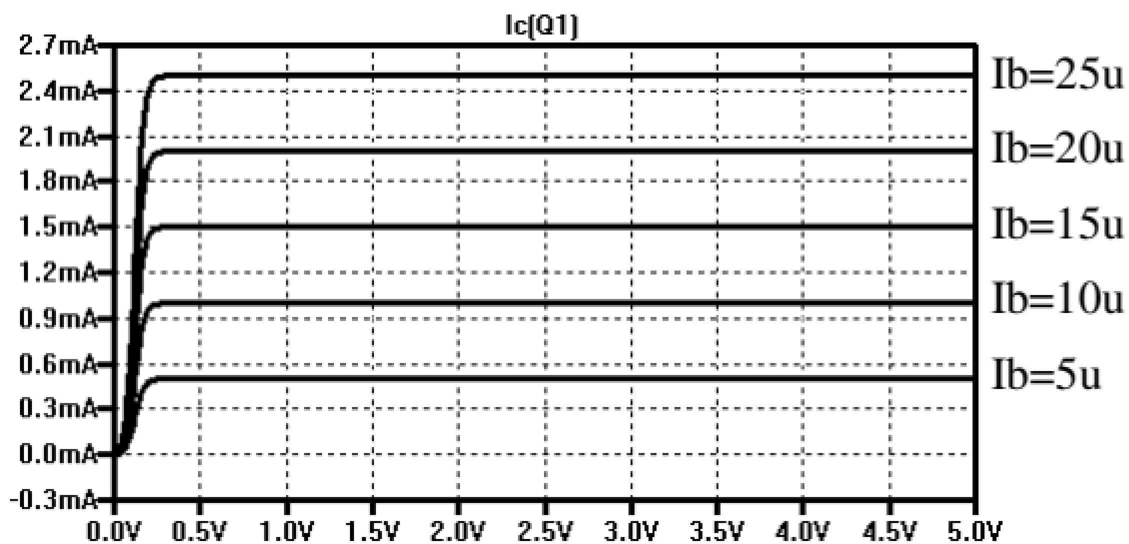

```
*#destroy all
*#run
*#let ID=-Vin#branch
*#plot ID
.dc Vin 0 1 1m
Vin Vin 0 DC 1
R1 Vin Vd 1k
D1 Vd 0 mydiode
.model mydiode D
.end
```



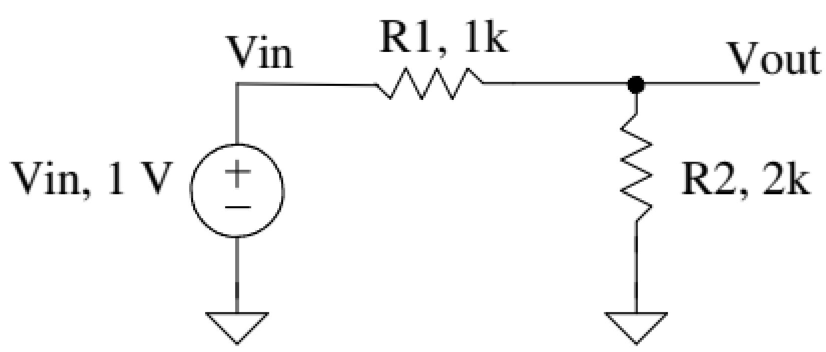
Example-09 (DC Sweep)



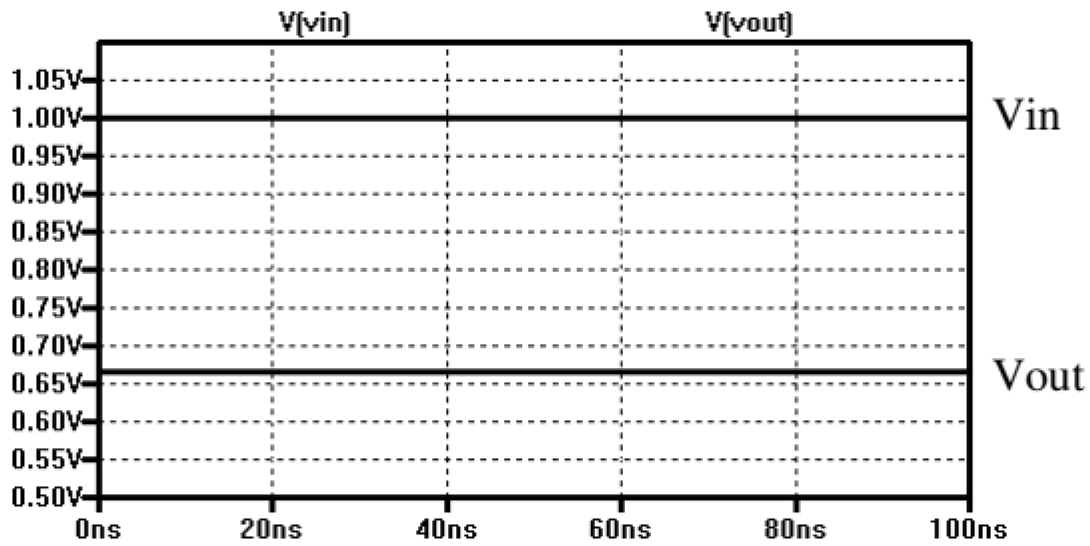

```
*#destroy all
*#run
*#let Ic=-Vce#branch
*#plot Ic
.dc Vce 0 5 1m Ib 5u 25u 5u
Vce Vce 0 DC 0
Ib 0 Vb DC 0
Q1 Vce Vb 0 myNPN
.model myNPN NPN
.end
```



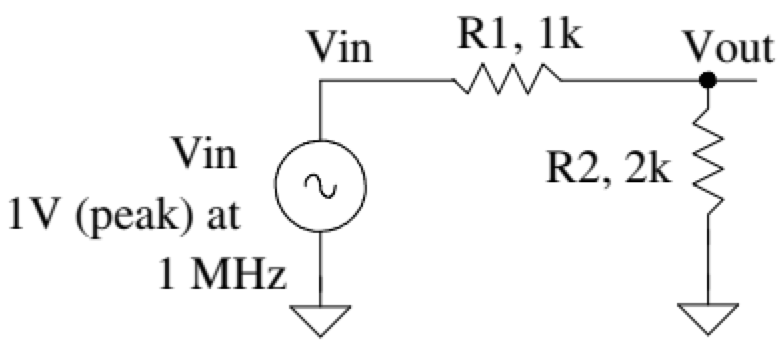
Example-10 (Transient Analysis)



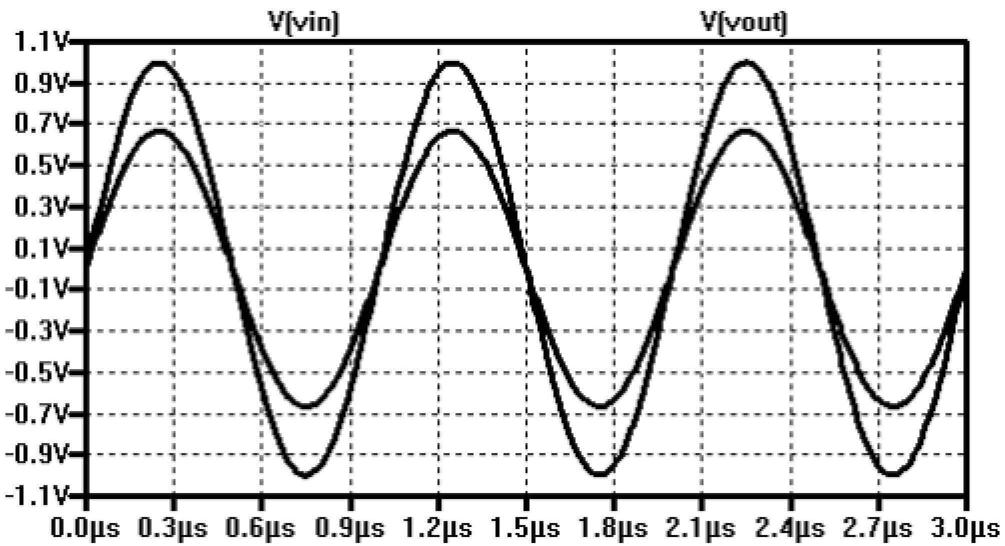

```
*#destroy all
*#run
*#plot vin vout

.tran 100p 100n

Vin     Vin     0       DC      1
R1      Vin     Vout    1k
R2      Vout    0       2k
.end
```



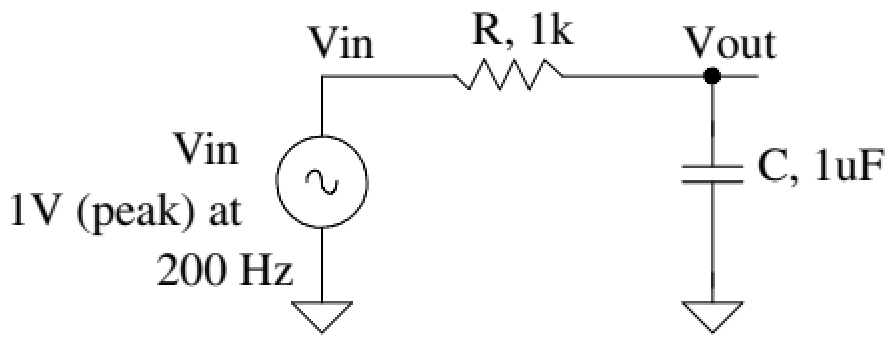
Example-11 (Transient Analysis)



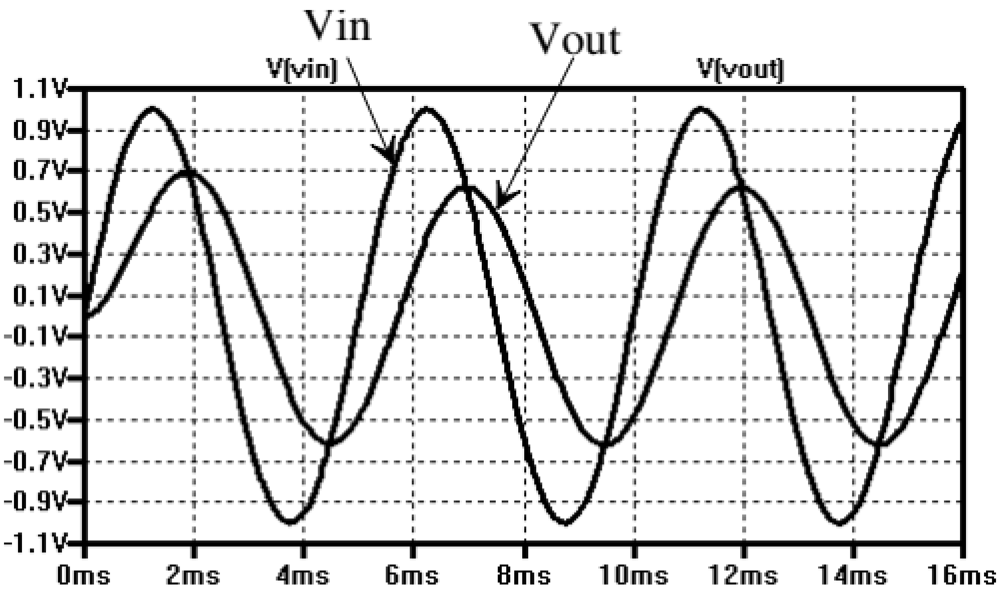

```
*#destroy all
*#run
*#plot vin vout
.tran 1n 3u
Vin Vin 0 DC 0 SIN 0 1 1MEG
R1 Vin Vout 1k
R2 Vout 0 2k
.end
```



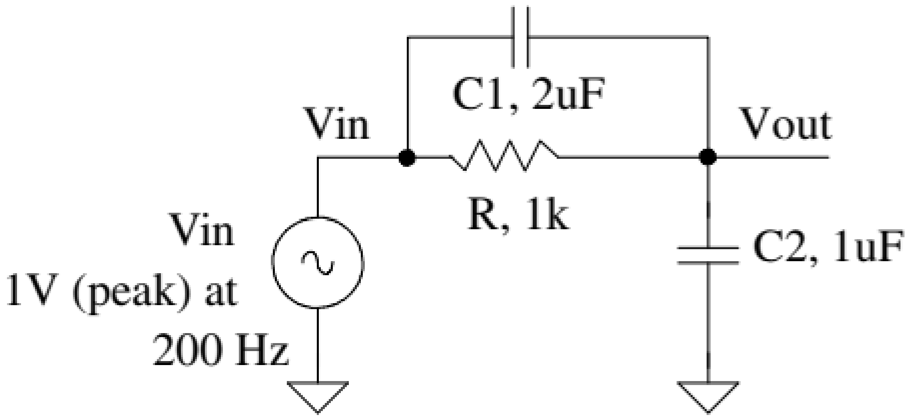
Example-12 (RC Circuit)



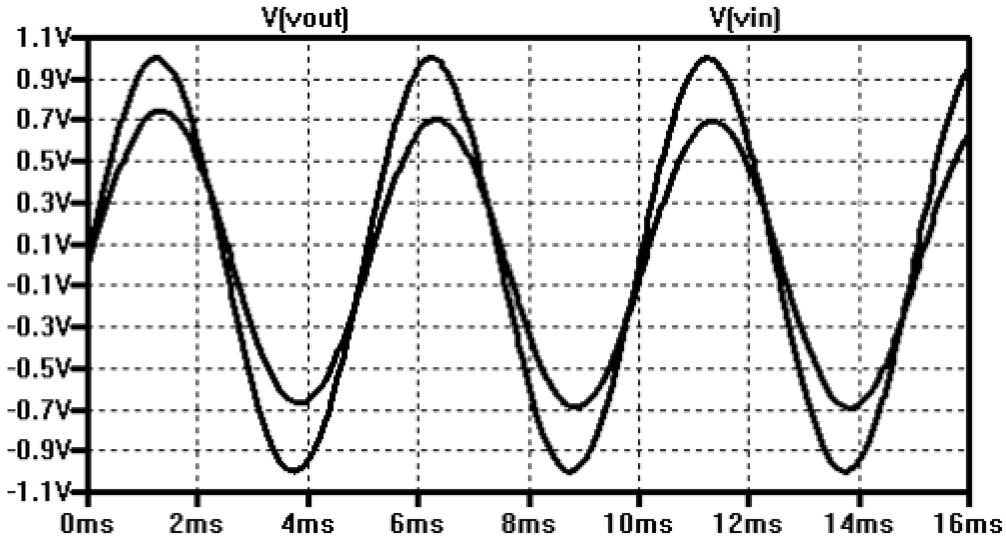

```
*#destroy all
*#run
*#plot vin vout
.tran 10u 16m
Vin Vin 0 DC 0 SIN 0 1 200
R1 Vin Vout 1k
CL Vout 0 1u
.end
```



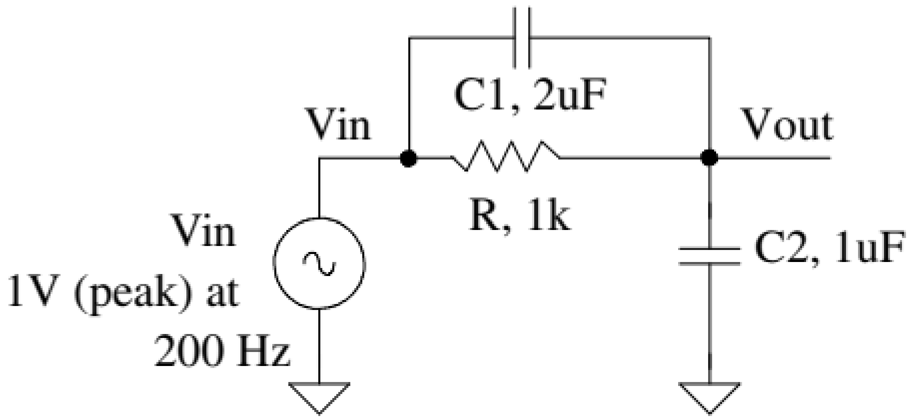
Example-13 (RC Circuit)




```
*#destroy all
*#run
*#plot vin vout
.tran 10u 16m
Vin Vin 0 DC 0 SIN 0 1 200
R1 Vin Vout 1k
C1 Vin Vout 2u
C2 Vout 0 1u
.end
```



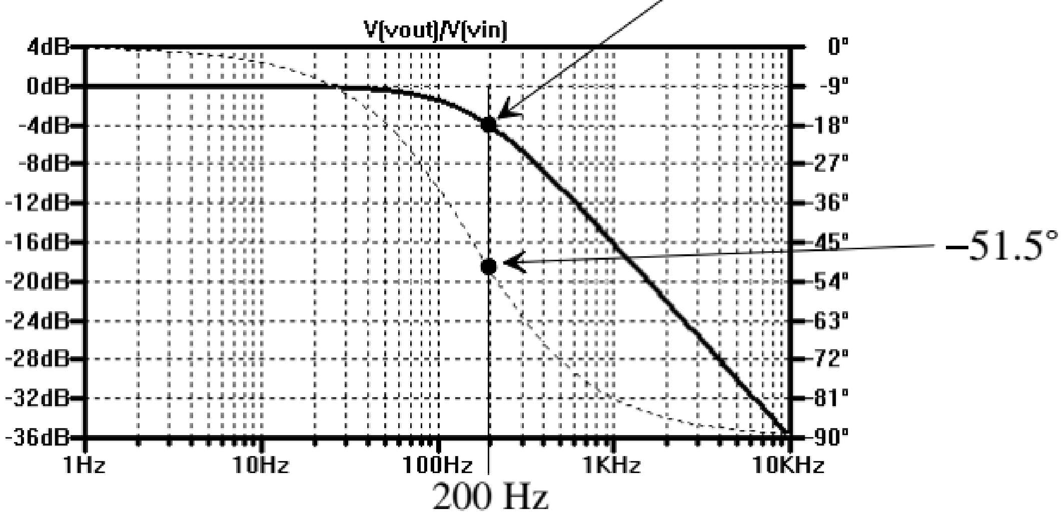
Example-14 (AC Analysis)




```
*#destroy all
*#run
*#plot db(vout/vin)
*#set units=degrees
*#plot ph(vout/vin)

.ac dec 100 1 10k

Vin     Vin     0       DC      0    SIN 0 1 200    AC 1
R1      Vin     Vout    1k
CL      Vout    0       1u

.end
```

.ac dec nd fstart fstop

$$20 \cdot \log(0.623) = -4.11\ dB$$



We can add a phase shift of 45 degrees by using AC 1 45 in the statement.

Important Terms (AC)

- **Decades** : Multiplying or dividing a frequency by 10
    - Example: One decade above 23 MHz is 230 MHz.
    - Example: One decade below 1.2 kHz is 120 Hz.
- **Octaves:** Multiplying or dividing a frequency by 2.
    - Example: One octave above 23 MHz is 46 MHz.
    - Example: One octave below 1.2 kHz is 600 Hz.
    - Two octaves above 23 MHz would be 92 MHz (multiply by 4).
- **Decibels (dB)Magnitude Changes:**
    - Decreasing the magnitude response by a factor of 10 corresponds to a drop of 20 dB.
    - Increasing the magnitude by a factor of 10 corresponds to a rise of 20 dB.

Important Terms (AC)

Frequency Roll-Off:

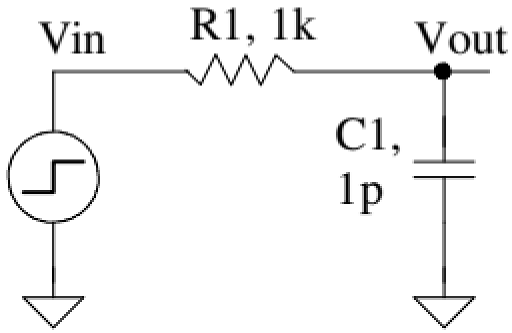
- For every increase in frequency by 10, the magnitude response decreases by 10.
- **Example** :Above 159 Hz, the response rolls off at 20 dB/decade.
- For every doubling (x2) in frequency, the magnitude response decreases by 2
- **Example :** response rolls off at 6 dB/octave above 159 Hz.

Comparison of Roll-Off Rates

- 6 dB/octave = 20 dB/decade.
- For a roll-off rate of 40 dB/decade, every frequency increase by 10 leads to a magnitude drop by 100.
- Similarly, at 12 dB/octave, doubling the frequency results in a magnitude drop by 4.

Example-15 (Pulse Statement)

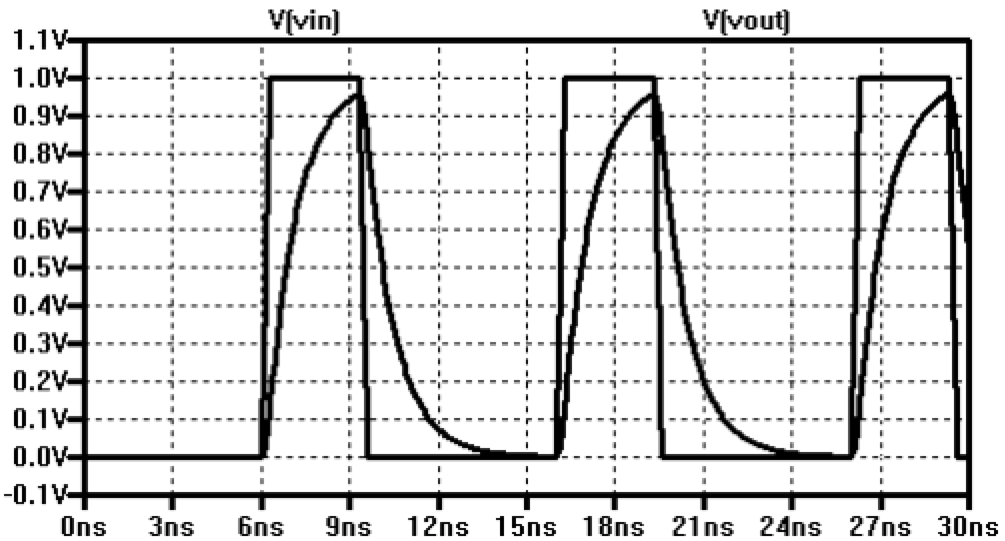
0 to 1 V
delay 6ns
time at 1 V = 3 ns
period = 10 ns



.tran 100p 30n

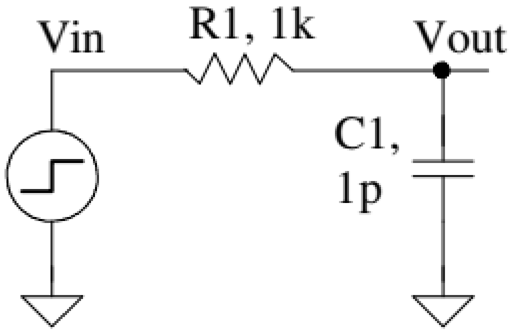
pulse vinit vfinal td tr tf pw per

Vin	Vin	0	DC	0	pulse 0 1 6n 0 0 3n 10n
R1	Vin	Vout	1k		
C1	Vout	0	1p		



Example-15 (Step Response –Positive Going)

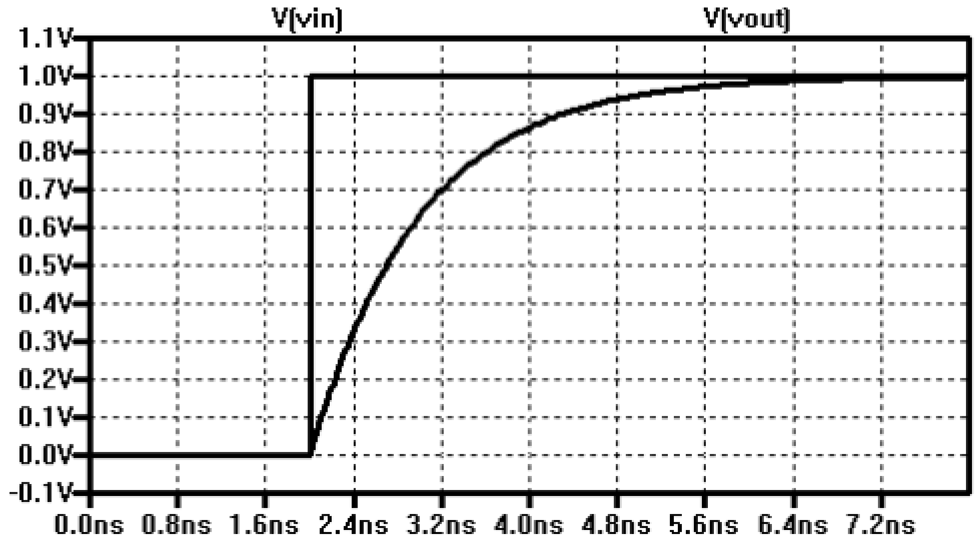
0 to 1 V
delay 6ns
time at 1 V = 3 ns
period = 10 ns



Vin Vin 0 DC 0 pulse 0 1 2n 10p

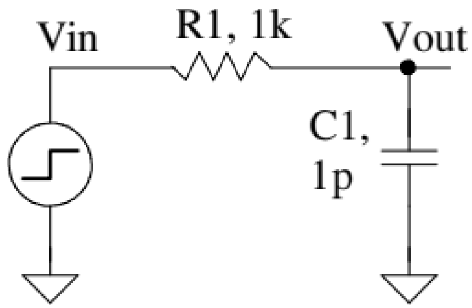
.tran 100p 30n

```
Vin   Vin    0     DC   0     pulse 0 1 6n 0 0 3n 10n
R1    Vin    Vout  1k
C1    Vout   0     1p
```



Example-15 (Step Response –Negative Going)

0 to 1 V
delay 6ns
time at 1 V = 3 ns
period = 10 ns



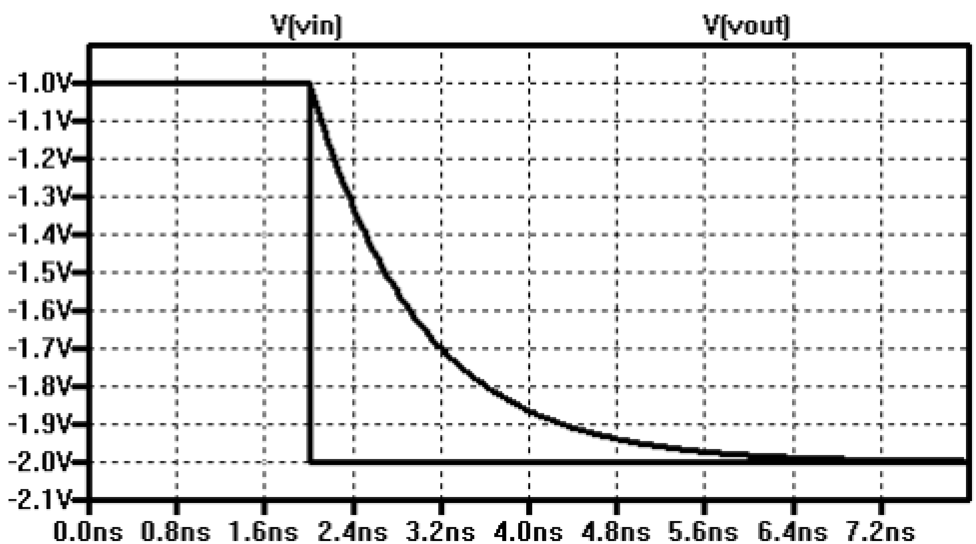
Vin Vin 0 DC 0 pulse -1 -2 2n 10p

$t_d \approx 0.7RC$

.tran 100p 30n

Vin	Vin	0	DC	0	pulse 0 1 6n 0 0 3n 10n
R1	Vin	Vout	1k		
C1	Vout	0	1p		

$t_r \approx 2.2RC$

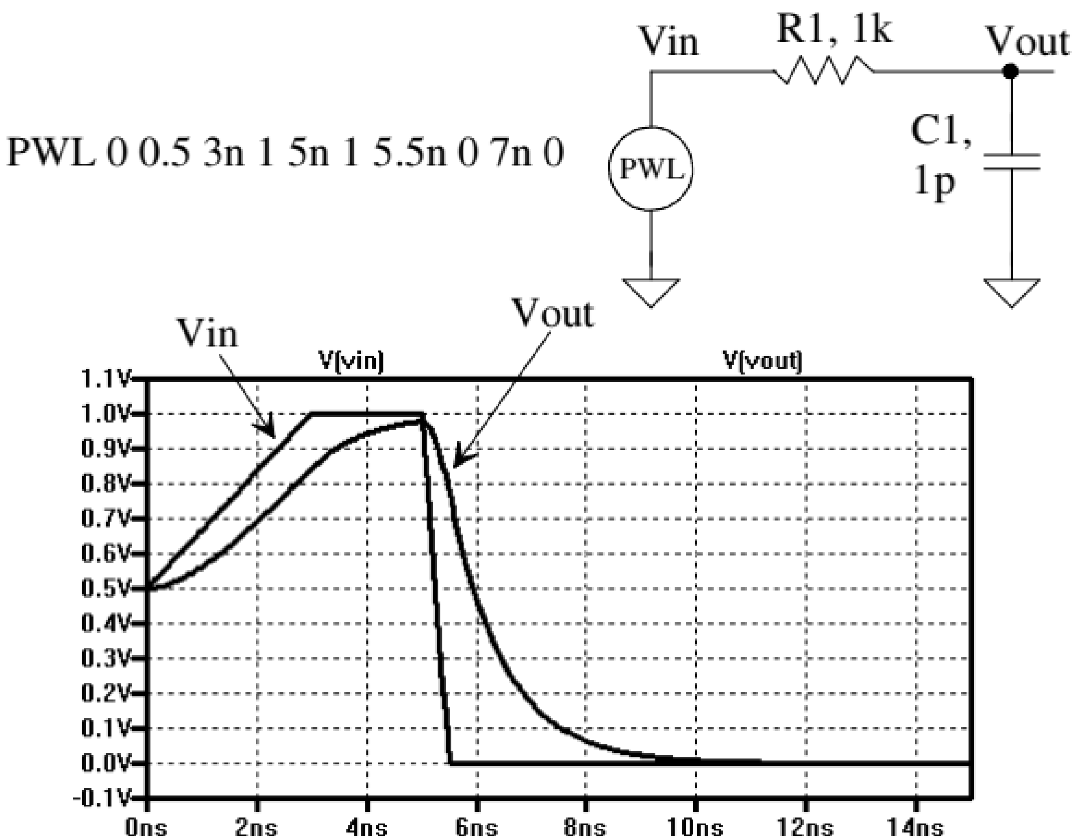


Example-16 (Piece-Wise Linear Source)

The piece-wise linear (PWL) source specifies arbitrary waveform shapes

```
pwl t1 v1 t2 v2 t3 v3 ... <rep>
```

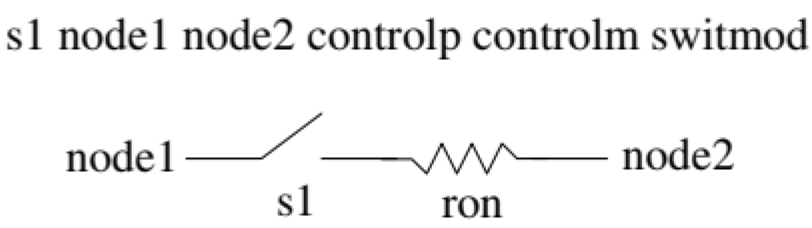
```
pwl 0 0.5 3n 1 5n 1 5.5n 0 7n 0
```



Example-17 (Switches)

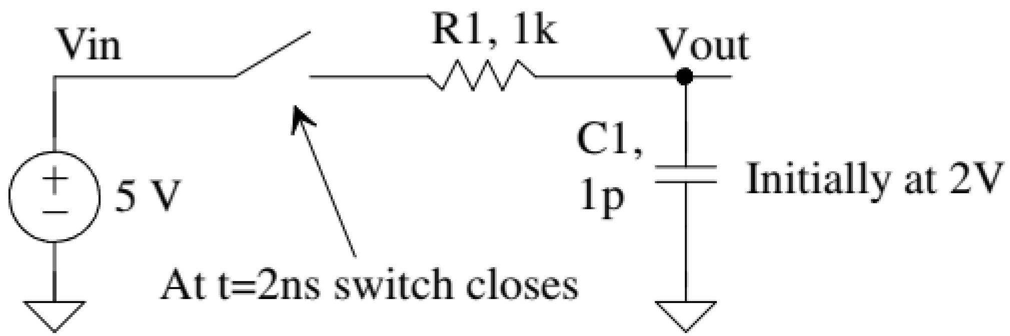
- The switch is closed when the node voltage **controlp** is greater than the node voltage **controlm**
- Switch is modeled using the .model statement
- On Series resistance of the switch to 1k can be set

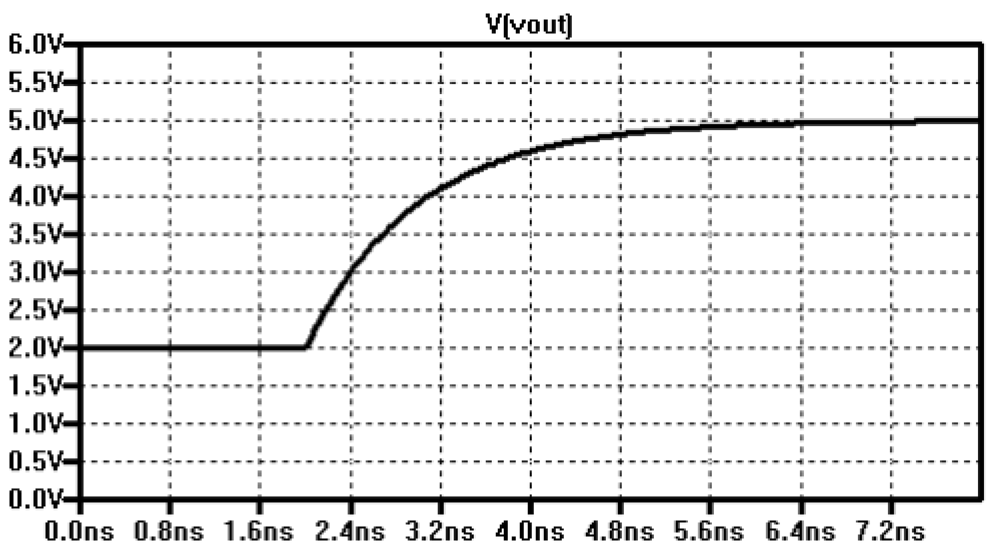
```
s1 node1 node2 controlp controlm switmod
.model switmod sw ron=1k
```



Example-18 (Initial Condition - Capacitor)

SPICE "use initial conditions" or skip an initial operating

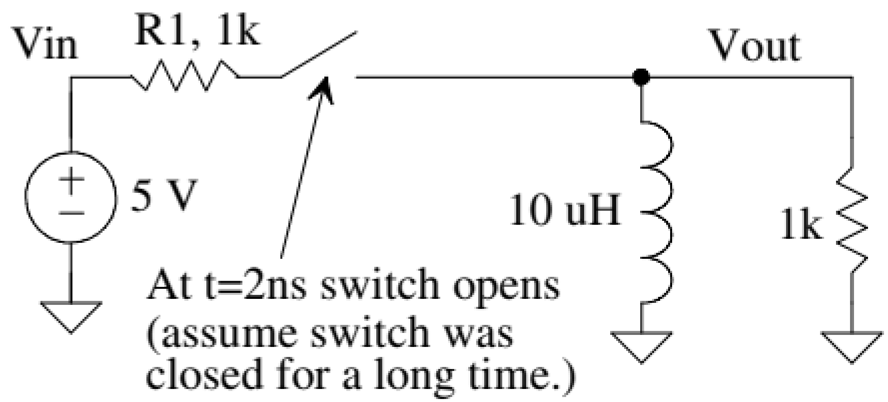


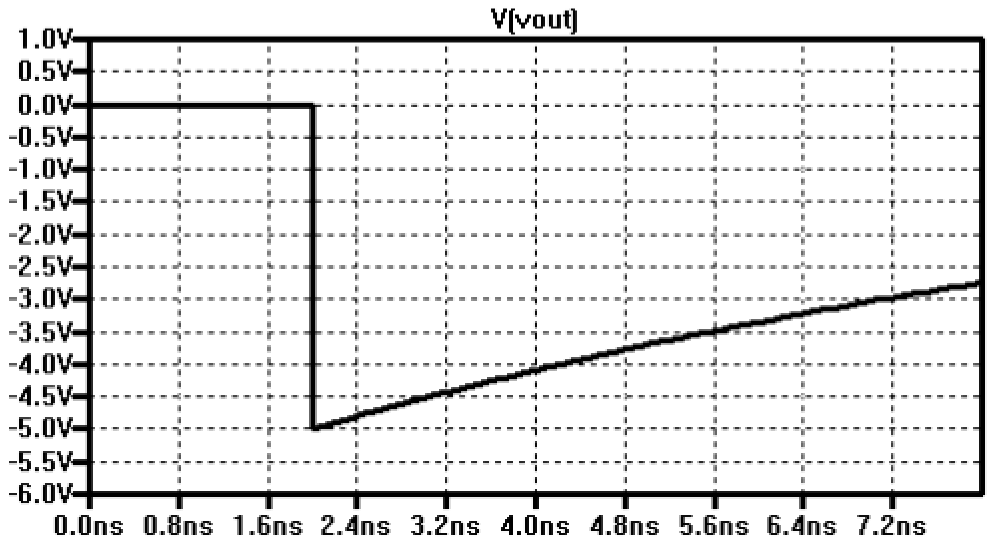



```
*#destroy all
*#run
*#plot vout

.tran 100p 8n UIC

Vclk clk 0 pulse -1 1 2n
Vin Vin 0 DC 5
S1 Vin Vouts clk 0 switmodel
R1 Vouts Vout 1k
C1 Vout 0 1p IC=2

.model switmodel sw ron=0.1

.end
```

.ic v(vout)=2

Example-19 (Initial Condition - Inductor)

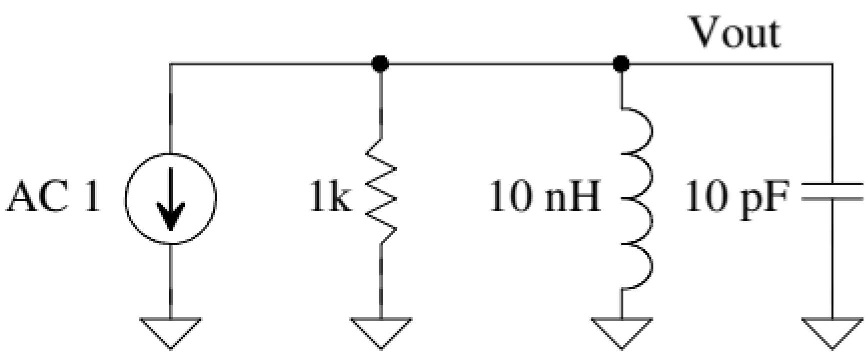


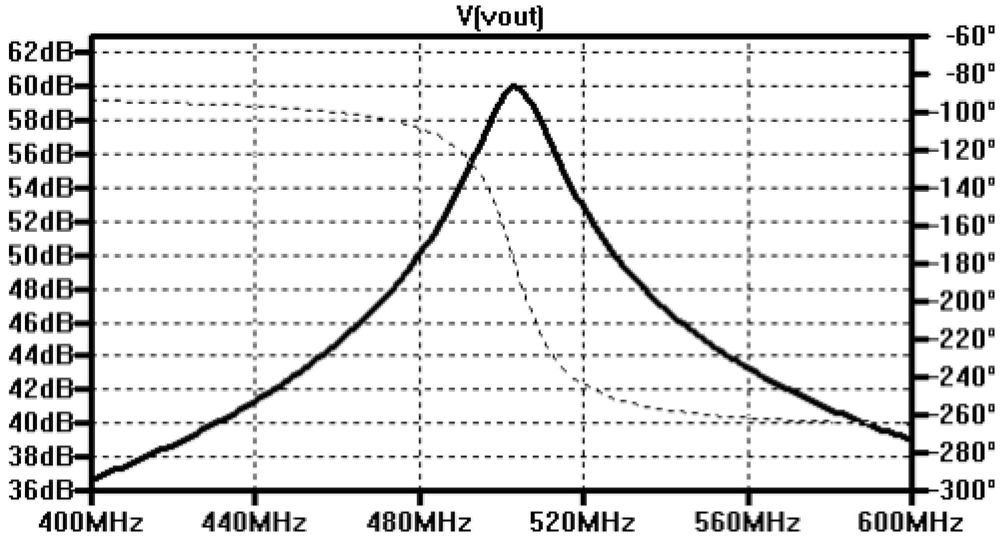



```
*#destroy all
*#run
*#plot vout

.tran 100p 8n UIC

Vclk clk 0 pulse -1 1 2n
Vin Vin 0 DC 5
R1 Vin Vouts1k
S1 Vouts Vout 0 clk switmodel
R2 Vout 0 1k
L1 Vout 0 10u IC=5m

.model switmodel sw ron=0.1

.end
```

Example-20 (Q of an LC Tank)

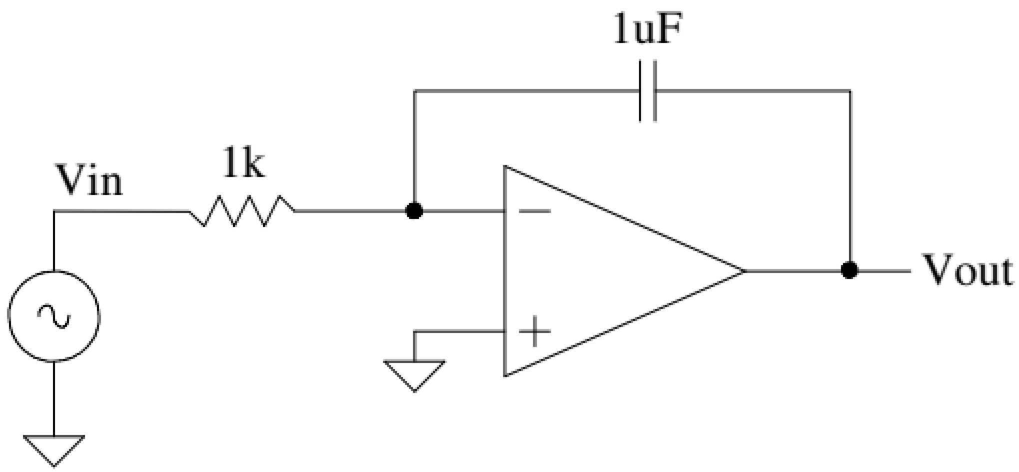
$$Q = \frac{f_{center}}{BW} = \frac{f_{center}}{f_{3dBhigh} - f_{3dBlow}}$$

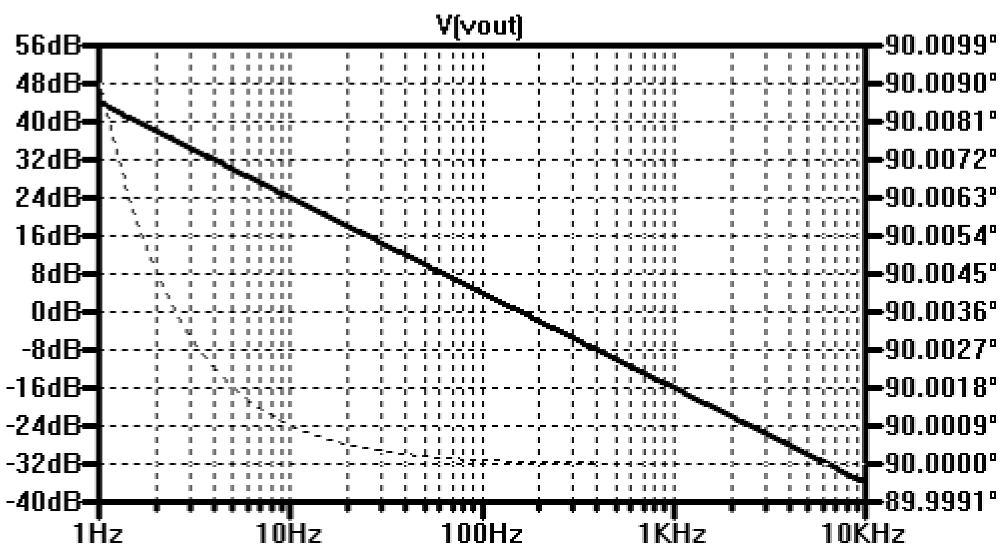





```
*#destroy all
*#run
*#plot db(vout)

.AC lin 100 400MEG 600MEG

Iin Vout 0 DC 0 AC 1
R1 Vout 0 1k
L1 Vout 0 10n
C1 Vout 0 10p

.end
```

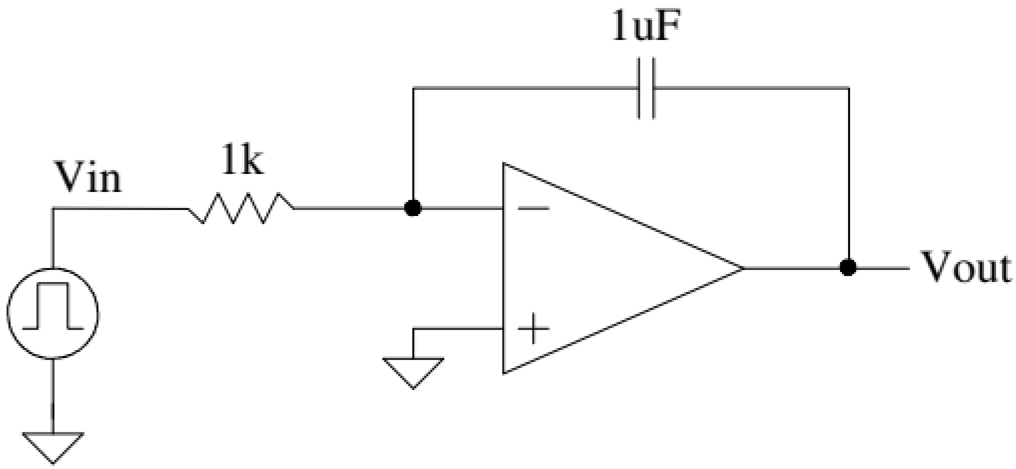
Example-21 (Integrator Response)

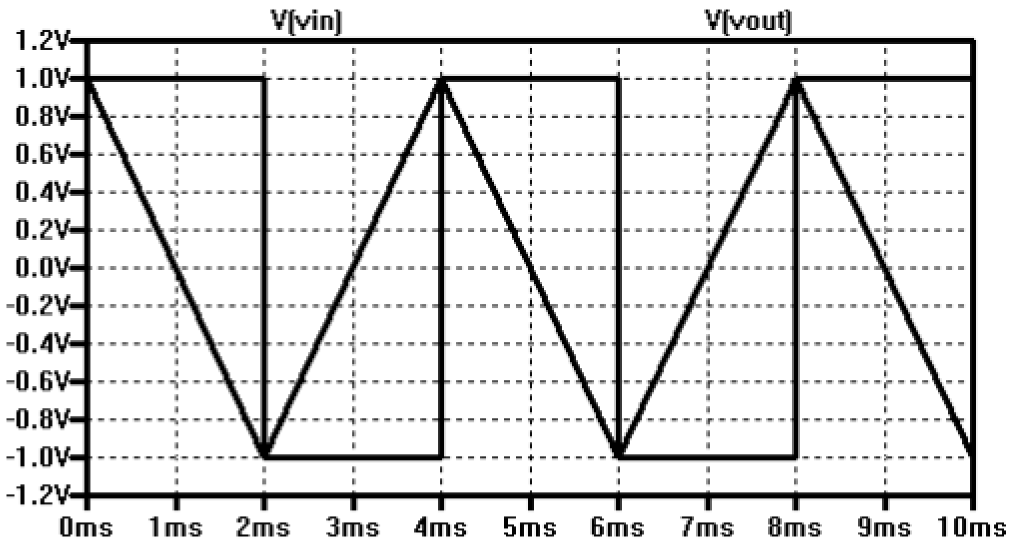





```
*#destroy all
*#run
*#plot db(vout/vin)
*#set units=degrees
*#plot ph(vout/vin)

.ac dec 100 1 10k

Vin Vin 0 DC 1 AC 1
Rin Vin vm 1k
Cf Vout vm 1u

X1 Vout 0 vm Ideal_op_amp
.subckt Ideal_op_amp Vout Vp Vm
G1 Vout 0 Vm Vp 1MEG
RL Vout 0 1
.ends
.end
```

Example-22 (Integrator Time Domain)





$$V_{out} = \frac{1}{C}\int \frac{V_{in}}{R} \cdot dt$$

$$V_{out}(t) = \frac{V_{in}}{RC} = \frac{1}{1\ ms}$$

```
*#destroy all
*#run
*#plot vout vin

.tran 10u 10m
.ic v(vout)=0

Vin Vin 0 DC 1
+ pulse -1 1 0 1u 1u 2m 4m
Rin Vin vm 1k
Cf Vout vm 1u

X1 Vout 0 vm Ideal_op_amp
.subckt Ideal_op_amp Vout Vp Vm
G1 Vout 0 Vm Vp 1MEG
RL Vout 0 1
.ends
.end
```

Summary

- **Key for Circuit Validation:** SPICE simulates circuit behavior, reducing errors before fabrication
- **Accurate Modeling:** Provides realistic predictions of performance and reliability
- **Enables Optimization:** Guides efficient design adjustments for power, speed, and area
- **Flexible Convergence:** Adjustable settings aid simulations of complex VLSI circuits.

Thank you !
Happy Learning