

# Lecture 5: SPICE

CSCE 6933/5933

## Advanced Topics in VLSI Systems

**Instructor:** Saraju P. Mohanty, Ph. D.

**NOTE:** The figures, text etc included in slides are borrowed from various books, websites, authors pages, and other sources for academic purpose only. The instructor does not claim any originality.



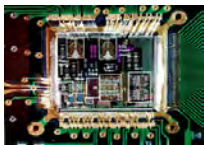
# Lecture Outlines

- About SPICE
- Supported analysis by SPICE
- Circuit Simulation
- Methods for solving linear and non linear equations



# About SPICE

- SPICE –Simulation Program with Integrated Circuit Emphasis
- SPICE is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses
- Circuits may contain:
  - resistors, capacitors, inductors, mutual inductors, independent voltage and current sources
  - four types of dependent sources
  - lossless and lossy transmission lines (two separate implementations), switches, uniform distributed RC lines
  - common semiconductor devices: diode, BJT, FET



# SPICE History

- 1969-70: Prof. Roher and a class project
  - CANCER: Computer Analysis of Nonlinear Circuits, Excluding Radiation
- 1970-72: Prof. Roher and Nagel
  - Develop CANCER into a truly public-domain, general-purpose circuit simulator
- 1972: SPICE I released as public domain
  - SPICE: Simulation Program with Integrated Circuit Emphasis
- 1975: Cohen following Nagel research
  - SPICE 2A released as public domain
- 1976 SPICE 2D New MOS Models
- 1979 SPICE 2E Device Levels (R. Newton appears)
- 1980 SPICE 2G Pivoting (ASV appears)



# SPICE History ...

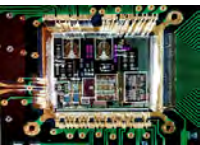
## Free Tools

- SPICE from Berkeley
- WinSpice
- NgSpice
- TclSpice
- LTSpice
- XSpice
- KSpice

## Commercial Tools

- Spectre
- PSpice
- HSpice
- TopSpice
- B2-Spice

Some of them run in Unix and some of them in Windows.



# SPICE History ...

- **S**imulation **P**rogram with **I**ntegrated **C**ircuit **E**mphasis
  - Developed in 1970's at Berkeley
  - Many commercial versions are available
  - HSPICE is a robust industry standard
    - Has many enhancements that we will use
- The circuits elements are called **cards**, and complete description is called a **SPICE deck**.
- The file contains a netlist consisting of components and nodes, simulation options, analysis options, and device models.



# Writing Spice Decks

- Writing a SPICE deck is like writing a good program
  - Plan: sketch schematic on paper or in editor
    - Modify existing decks whenever possible
  - Code: strive for clarity
    - Start with name, email, date, purpose
    - Generously comment
  - Test:
    - Predict what results should be
    - Compare with actual
    - *Garbage In, Garbage Out!*



# SPICE Sources

- DC Source

Vdd vdd gnd 2.5

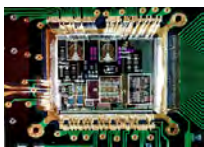
- Piecewise Linear Source

Vin in gnd pwl 0ps 0 100ps 0 150ps 1.8 800ps 1.8

- Pulsed Source

Vck clk gnd PULSE 0 1.8 0ps 100ps 100ps 300ps 800ps

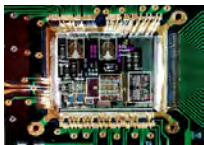
**PULSE v1 v2 td tr tf pw per**





# SPICE Elements

Letter	Element
R	Resistor
C	Capacitor
L	Inductor
K	Mutual Inductor
V	Independent voltage source
I	Independent current source
M	MOSFET
D	Diode
Q	Bipolar transistor
W	Lossy transmission line
X	Subcircuit
E	Voltage-controlled voltage source
G	Voltage-controlled current source
H	Current-controlled voltage source
F	Current-controlled current source



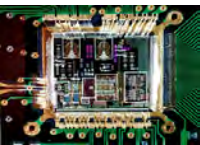
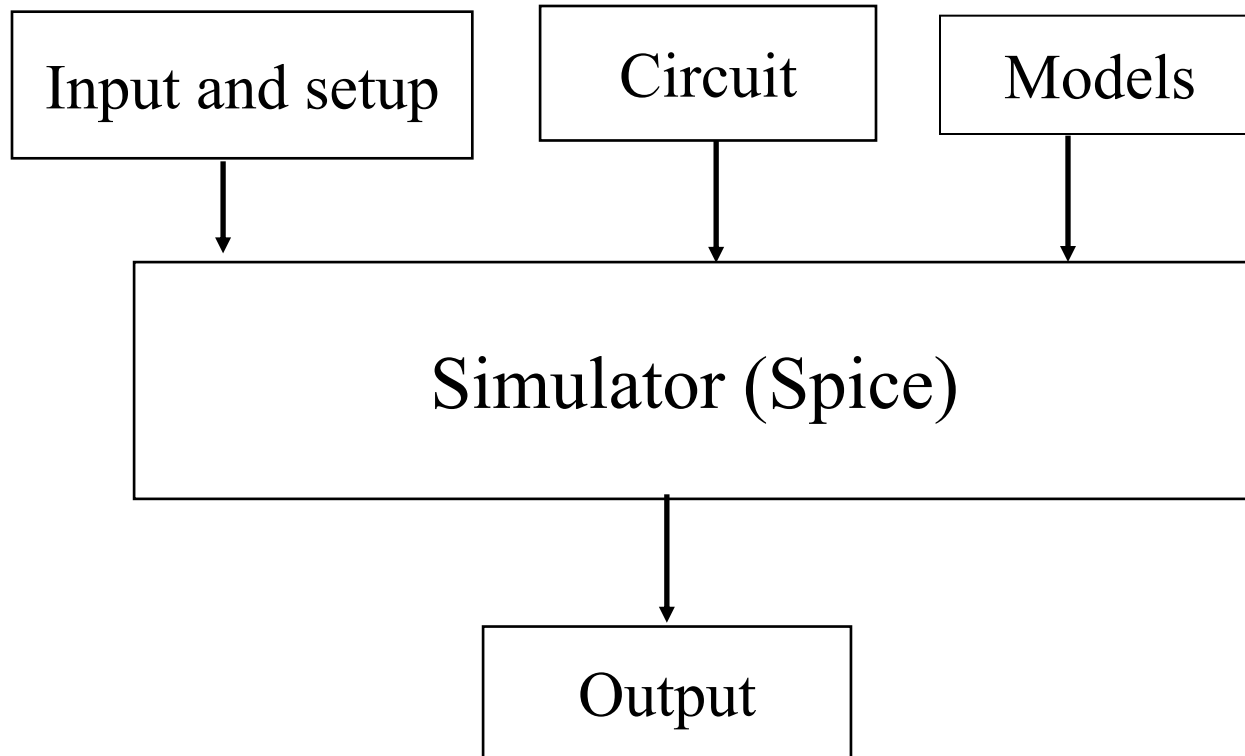
# SPICE Units

Letter	Unit	Magnitude
a	atto	$10^{-18}$
f	femto	$10^{-15}$
p	pico	$10^{-12}$
n	nano	$10^{-9}$
u	micro	$10^{-6}$
m	mili	$10^{-3}$
k	kilo	$10^3$
x	mega	$10^6$
g	giga	$10^9$

Ex: 100 femptofarad capacitor = 100fF, 100f, 100e-15



# Circuit Simulation: User's Perspective



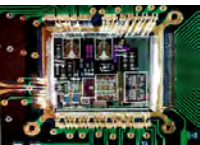
# Simulators at Different Levels

- Simulators at different levels of abstraction :
  - Process
  - Circuit
  - Logic
- **Process Simulator**: Time and temperature effect on physical and electrical characteristics.
- **Circuit Simulator**: Use device model and netlist to predict current, voltage, and performance.
- **Logic Simulator**: Functional prediction of digital circuits for logical correctness.



# Device Models

- SPICE uses wide variety of transistor models with different trade-offs between complexity and accuracy.
- Different models:
  - Level 1 (simple)
  - Level 2 (simple)
  - Level 3 (simple)
  - BSIM 1 (Berkley Short-Channel Insulated-Gate-FET Model)
  - BSIM 2 (very elaborate)
  - BSIM 3 (very elaborate)
  - BSIM 4 (current, most elaborate)

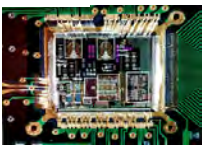


# Device Models : Level 1

- Closely related to the Shockley model
- Enhanced with channel length modulation and the body effect.
- The parameters from the SPICE model are in CAPS.
- The threshold voltage is modulation by the source-to-body voltage ( $V_{sb}$ ) due to body effect is taken into account.
- Gate capacitance is calculated from the oxide thickness TOX.

```
.model NMOS NMOS (LEVEL=1 TOX=40e-10 KP=155E-6 LAMBDA=0.2  
+           VTO=0.4 PHI=0.93 GAMMA=0.6  
+           CJ=9.8E-5 PB=0.72 MJ=0.36  
+           CJSW=2.2E-10 PHP=7.5 MJSW=0.1)
```

Sample Level 1 MODEL card



# Device Models : Level 2 and Level 3

- Account effects of :
  - Velocity Saturation
  - Mobility Degradation
  - Subthreshold Conduction
  - Drain-Induced Barrier Lowering
- Still not good enough to mimic the characteristics of the modern devices.



# Device Models : BSIM

- Features of version 3v3:
  - Continuous and differentiable I-V characteristics for different regions of operation for good convergence.
  - Sensitivity of  $V_t$ , etc to length and width
  - $V_t$  model include body effect and DIBL.
  - Velocity saturation, mobility degradation, etc.
  - Multiple gate capacitance
  - Diffusion capacitance and resistance models
- Features of version 4:
  - Tunneling current
  - High-K dielectric





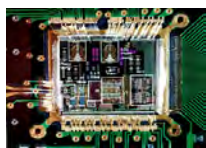
# Analysis Supported by SPICE

- Types of Analysis
  - DC Analysis
  - AC Small-Signal Analysis
  - Transient Analysis
  - Pole-Zero Analysis
  - Small-Signal Distortion Analysis
  - Sensitivity Analysis
  - Noise Analysis



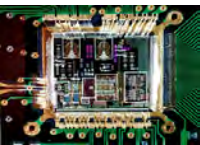
# Analysis Supported by SPICE

COMMON SPICE ANALYSES	
.AC	AC Analysis
.DC	DC Analysis
.FOUR	Fourier Analysis
.NOISE	Small-Signal Noise Analysis
.OP	Operating Point Analysis
.SENS	Sensitivity Analysis
.TRAN	Transient Analysis
ANALYSES SPECIFIC TO SPICE2G6	
.DISTO	Small-Signal Distortion Analysis
ANALYSES SPECIFIC TO SPICE3	
.DISTO	Small-Signal Distortion Analysis
.PZ	Pole-Zero Analysis
ANALYSES SPECIFIC TO PSpICE	
.TF	Transfer Function Specification PSpICE Only
.MC	Monte Carlo Analysis (PSpICE only)
.SAVEBIAS	Save Bias Conditions
.STEP	Parametric Analysis
.WCASE	Sensitivity and Worst Case Analysis



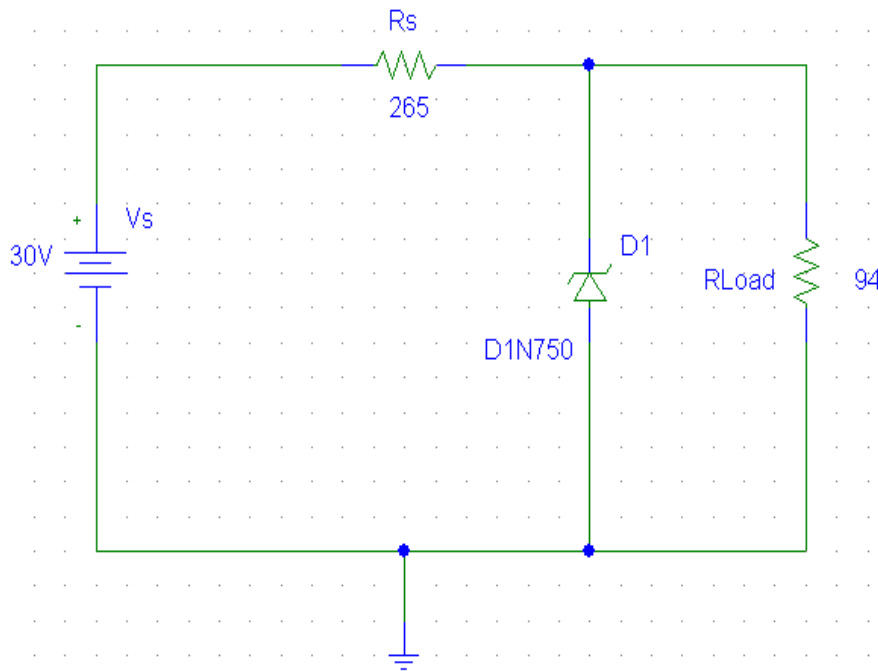
# Analysis : DC

- The dc analysis portion of SPICE determines the dc operating point of the circuit with inductors shorted and capacitors opened.
- The dc analysis options are specified on the .DC, .TF, and .OP control lines.
- A dc analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an ac small-signal analysis to determine the linearized, small-signal models for nonlinear devices.



# Analysis: DC, Example

Question: Voltage across RLoad as Vs varies from 20V to 30V.

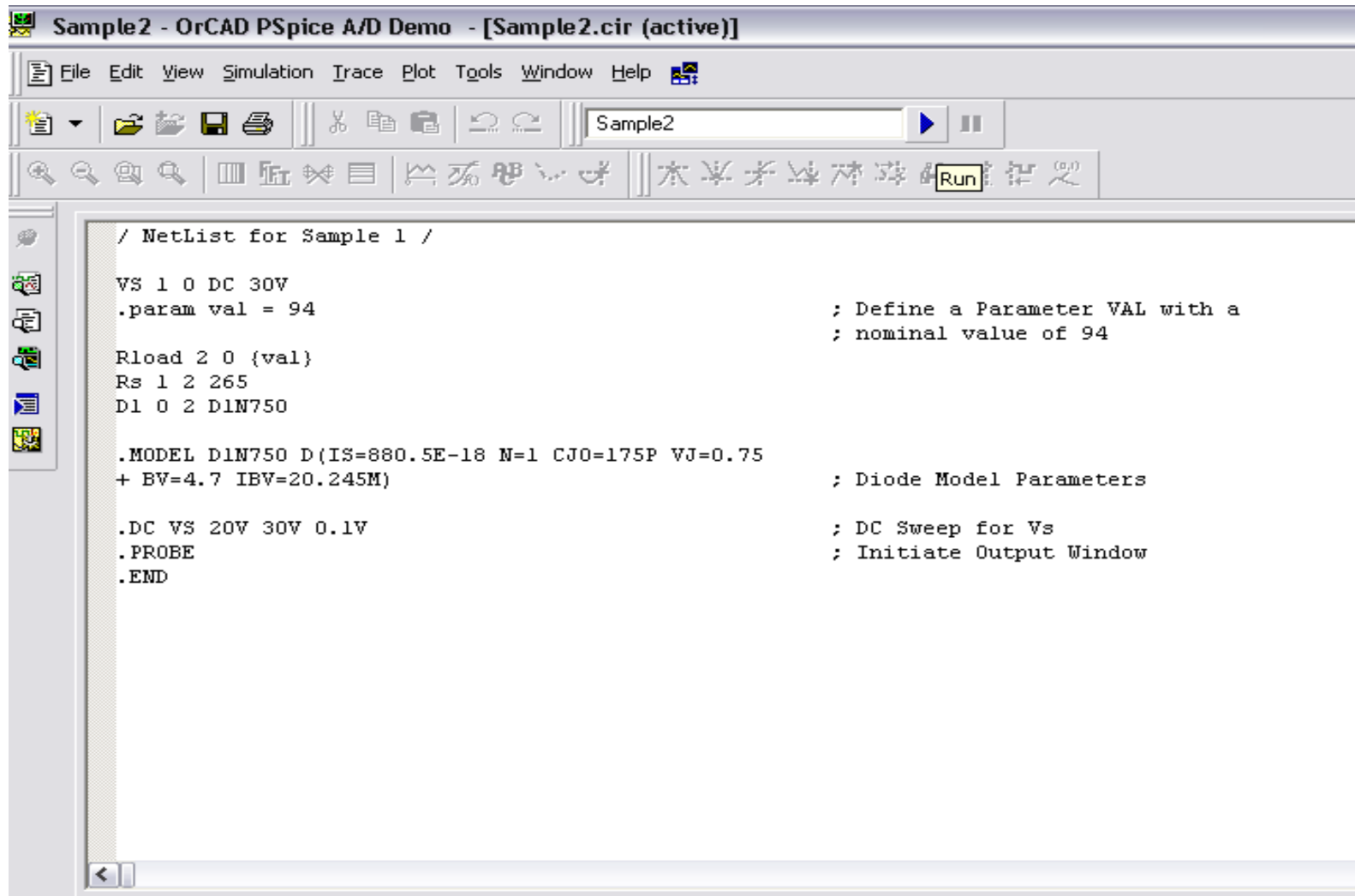


What is the voltage across RLoad as  $V_s$  varies from 20V to 30V?

This circuit can be analyzed using Schematic and Net list [Manual Coding].

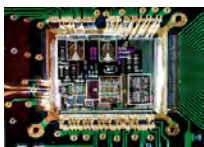


# Analysis: DC, Example ...

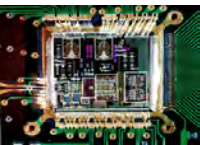
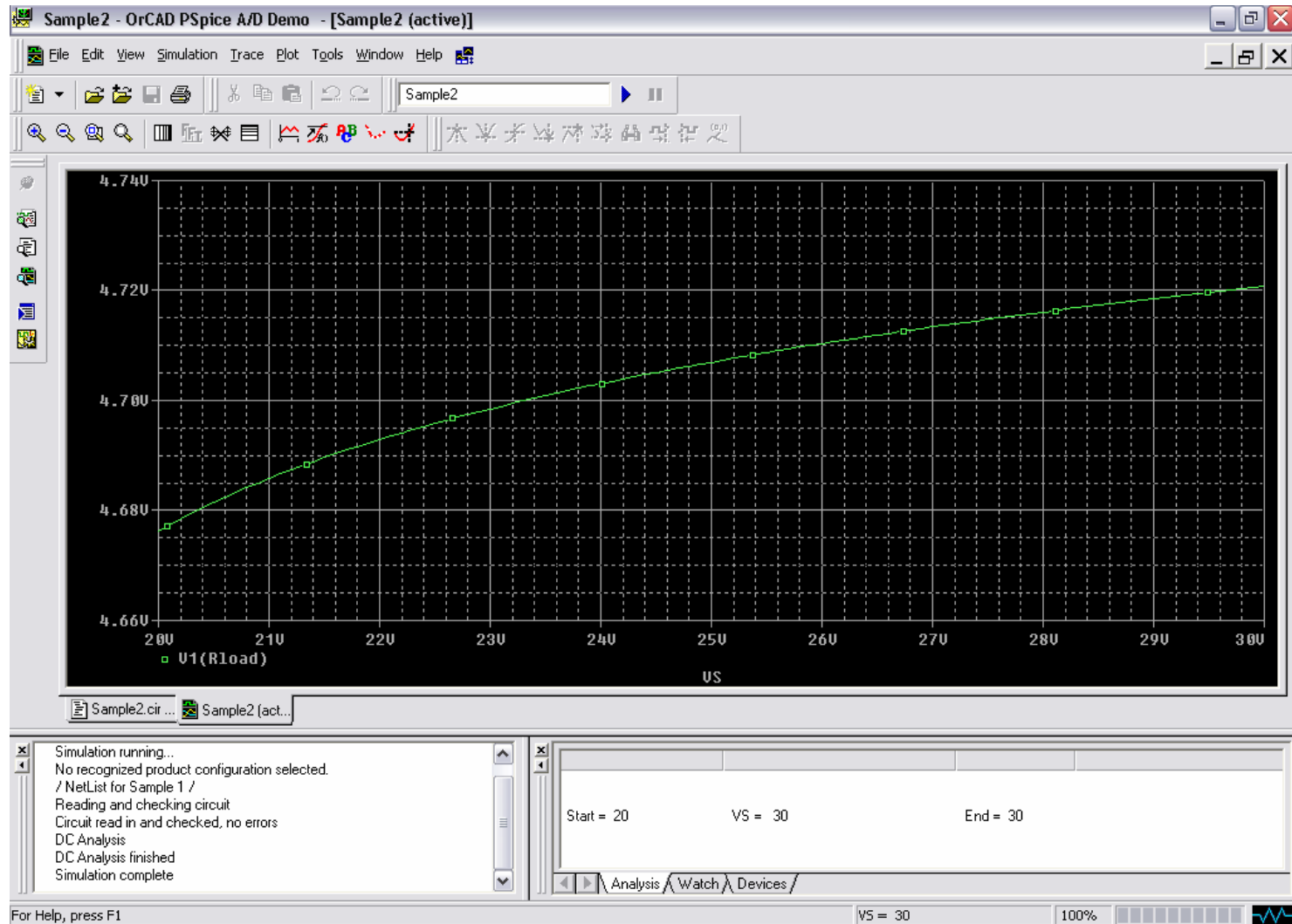


The screenshot shows the OrCAD PSpice A/D Demo software interface. The title bar reads "Sample2 - OrCAD PSpice A/D Demo - [Sample2.cir (active)]". The menu bar includes File, Edit, View, Simulation, Trace, Plot, Tools, Window, and Help. The toolbar contains various icons for file operations, simulation, and analysis. The main window displays a NetList for "Sample 1" with the following content:

```
/ NetList for Sample 1 /  
  
VS 1 0 DC 30V  
.param val = 94 ; Define a Parameter VAL with a  
; nominal value of 94  
  
Rload 2 0 {val}  
Rs 1 2 265  
D1 0 2 D1N750  
  
.MODEL D1N750 D(IS=880.5E-18 N=1 CJO=175P VJ=0.75  
+ BV=4.7 IBV=20.245M) ; Diode Model Parameters  
  
.DC VS 20V 30V 0.1V ; DC Sweep for Vs  
.PROBE ; Initiate Output Window  
.END
```

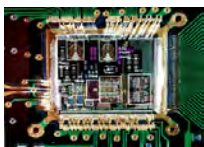
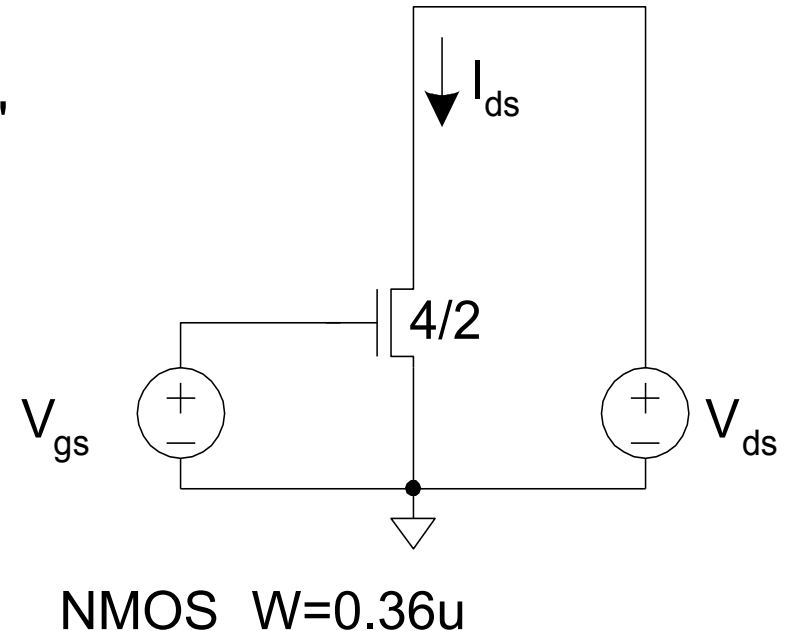


# Analysis: DC, Example ...



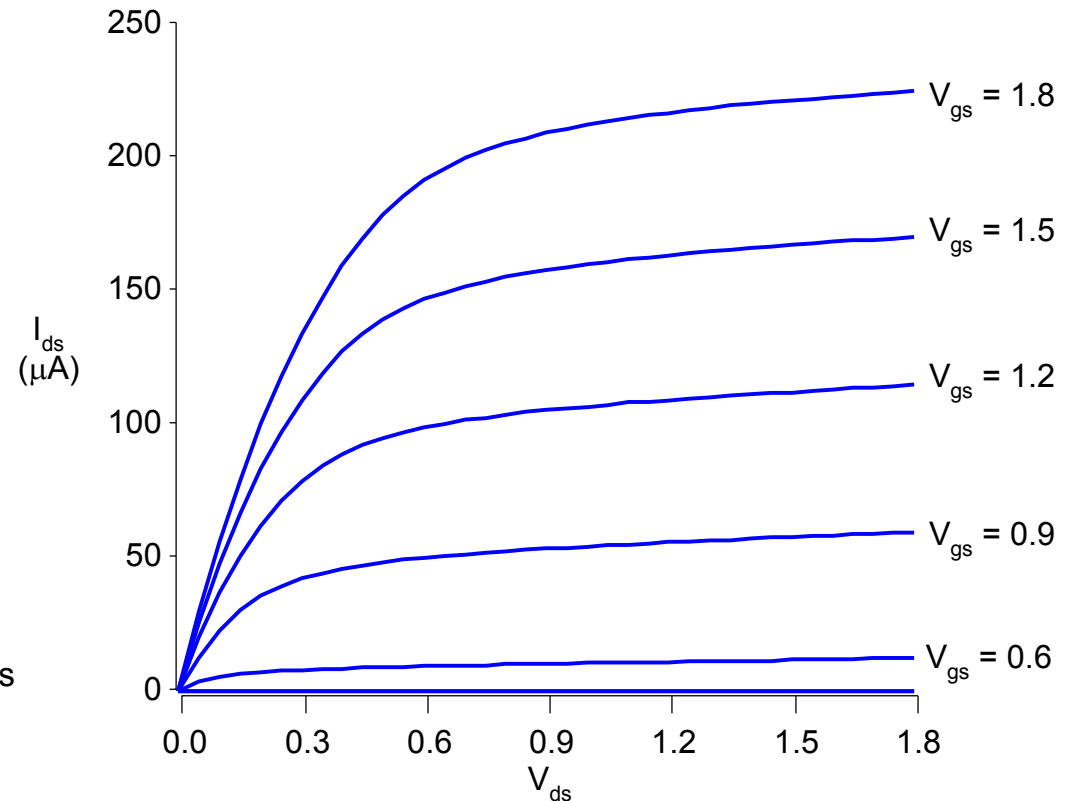
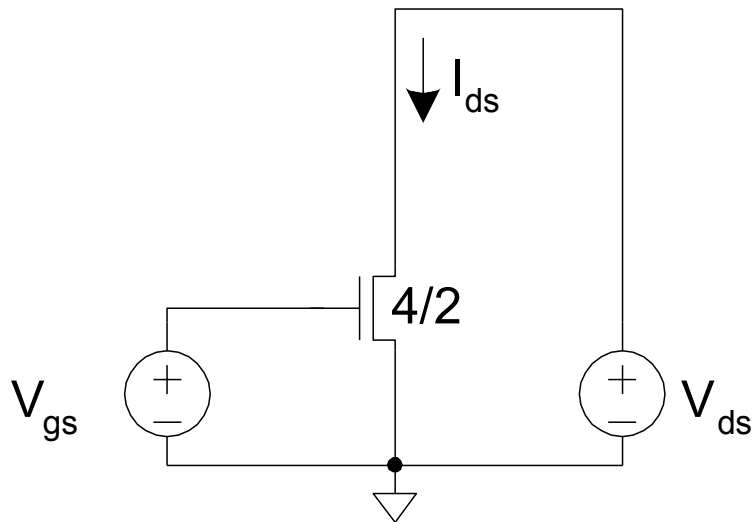
# Transistor DC Analysis

```
* mosiv.sp
* -----
* Parameters and models
* -----
.include '../models/tsmc180/models.sp'
.temp 70
.option post
* -----
* Simulation netlist
* -----
*nmos
Vgs      g      gnd      0
Vds      d      gnd      0
M1       d      g      gnd      gnd
          L=0.18u
* -----
* Stimulus
* -----
.dc Vds 0 1.8 0.05 SWEEP Vgs 0 1.8 0.3
.end
```

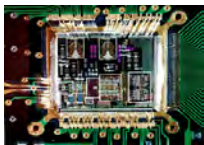


# Transistor DC Analysis

- NMOS I-V
  - $V_{gs}$  dependence
  - Saturation



I-V characteristics





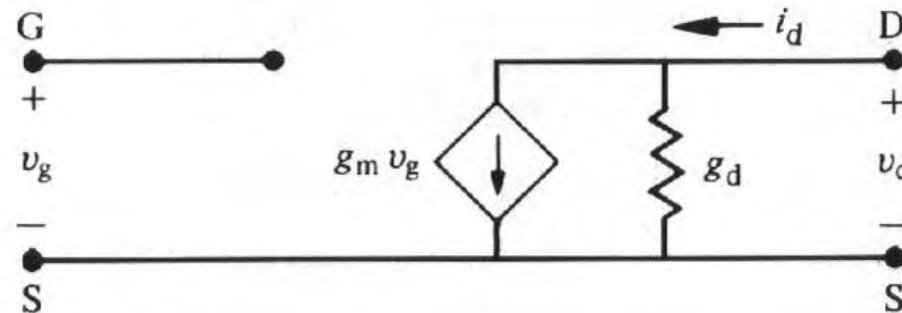
# Analysis: AC Small Signal

- The ac small-signal portion of SPICE computes the ac output variables as a function of frequency.
- The ac analysis options are specified on the .AC control lines.
- The program first computes the dc operating point of the circuit and determines linearized, small-signal models for all of the nonlinear devices in the circuit.

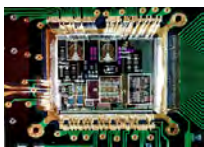
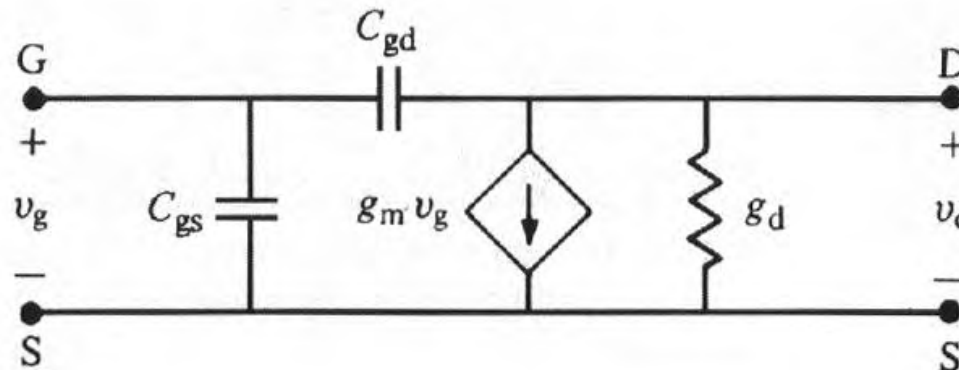


# Analysis: AC Small Signal of MSFET

MOSFET equivalent circuit at low-frequency after neglecting the gate capacitance for low frequency:

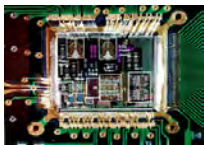


MOSFET equivalent circuit at high-frequency which includes the capacitive effects:



# Analysis: Transient

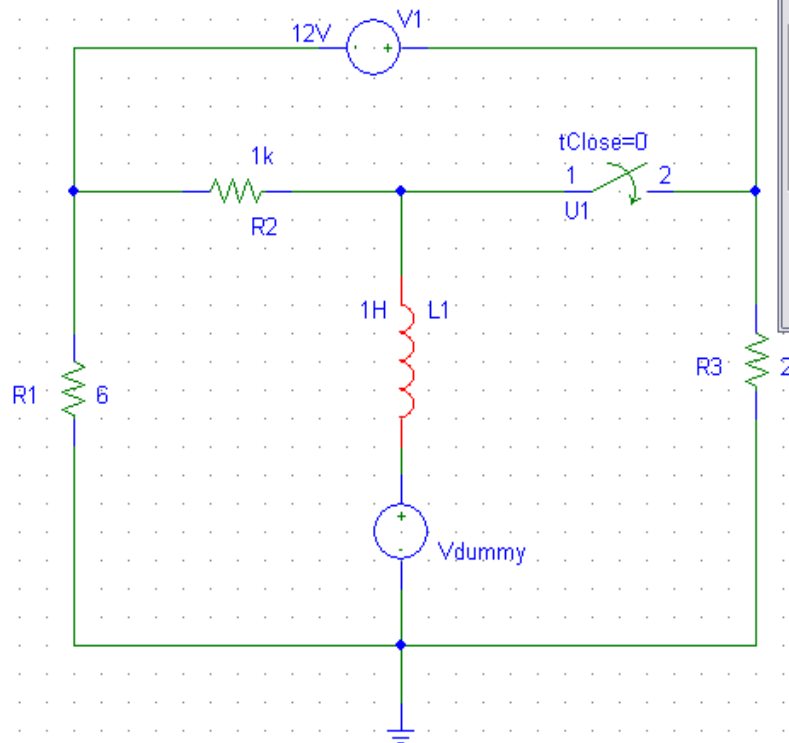
- The transient analysis portion of SPICE computes the transient output variables as a function of time over a user-specified time interval.
- The transient time interval is specified on a .TRAN control line.
- The initial conditions are automatically determined by a dc analysis.



# Analysis: Transient, Example

Question: Find  $V_{out}$  at  $t > 0$

Do Hand Calculation for current through Inductor at  $t < 0$

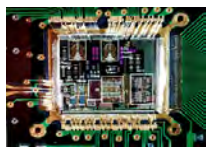


L1 PartName: L

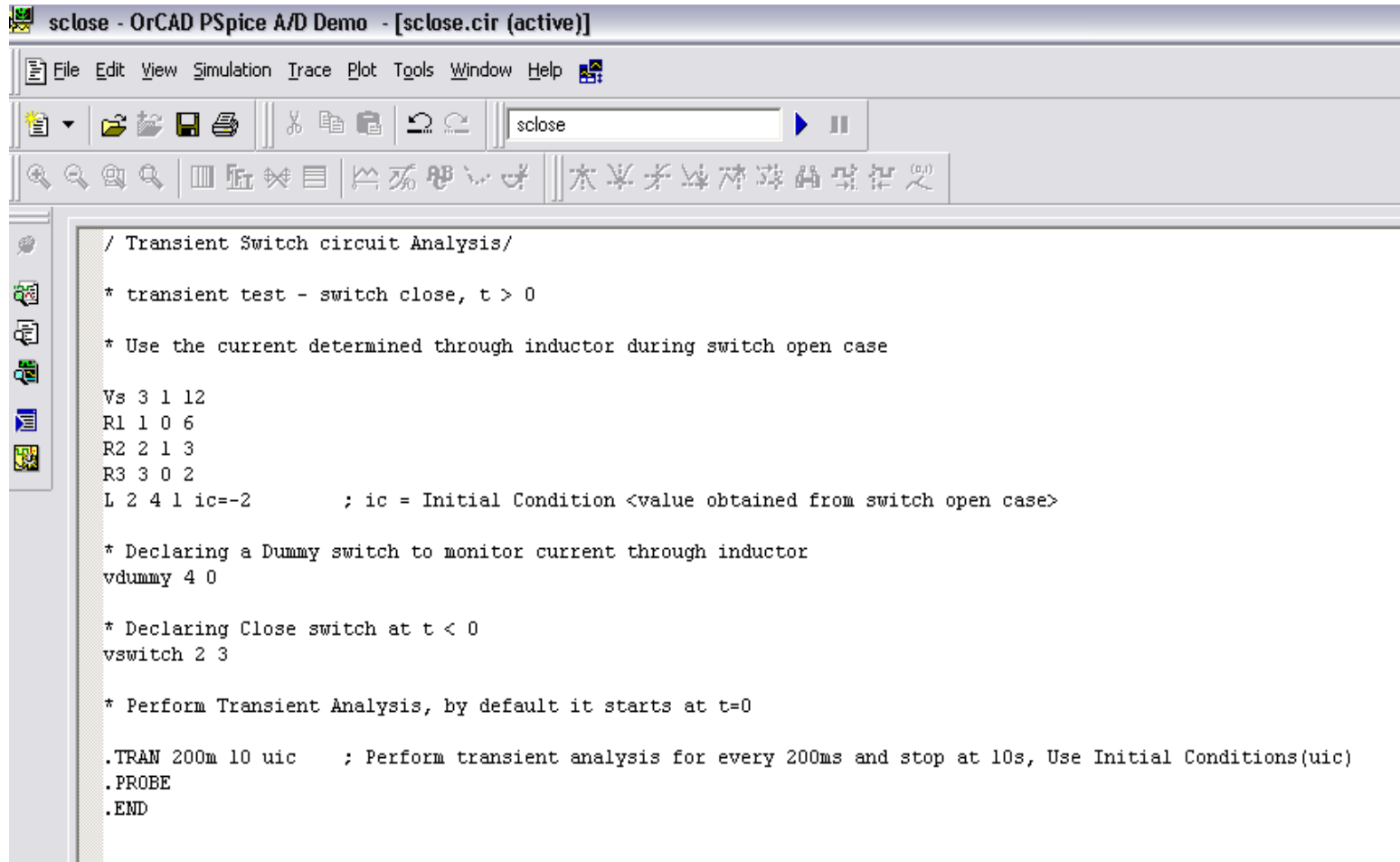
Name	Value
IC	= -2
VALUE	= 1H
IC	= -2
* REFDES	= L1
* TEMPLATE	= L^@REFDES %1 %2 ?TOLERANCE L^@REFDES
* PART	= L
TOLERANCE	=
PKGTYPE	= L2012C

Include Non-changeable Attributes  
 Include System-defined Attributes

Buttons: Save Attr, Change Display, Delete, OK, Cancel

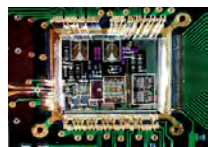


# Analysis: Transient, Example

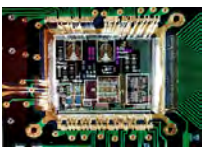
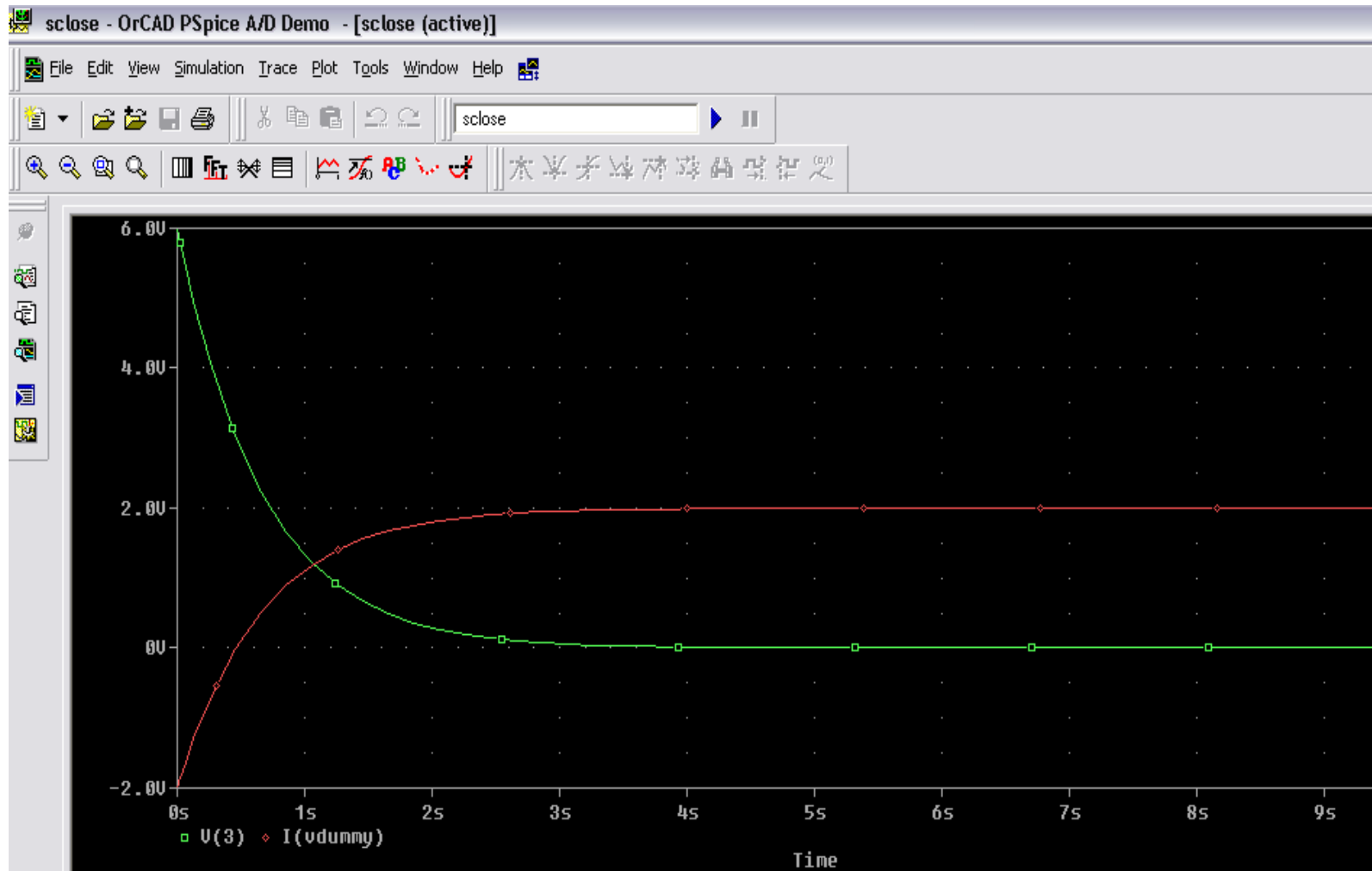


The screenshot shows the OrCAD PSpice A/D Demo interface. The title bar reads "sclose - OrCAD PSpice A/D Demo - [sclose.cir (active)]". The menu bar includes File, Edit, View, Simulation, Trace, Plot, Tools, Window, and Help. The toolbar contains various icons for file operations, simulation control, and analysis. The main window displays the following text:

```
/ Transient Switch circuit Analysis/  
  
* transient test - switch close, t > 0  
  
* Use the current determined through inductor during switch open case  
  
Vs 3 1 12  
R1 1 0 6  
R2 2 1 3  
R3 3 0 2  
L 2 4 1 ic=-2      ; ic = Initial Condition <value obtained from switch open case>  
  
* Declaring a Dummy switch to monitor current through inductor  
vdummy 4 0  
  
* Declaring Close switch at t < 0  
vswitch 2 3  
  
* Perform Transient Analysis, by default it starts at t=0  
  
.TRAN 200m 10 uic      ; Perform transient analysis for every 200ms and stop at 10s, Use Initial Conditions(uic)  
.PROBE  
.END
```



# Analysis: Transient, Example

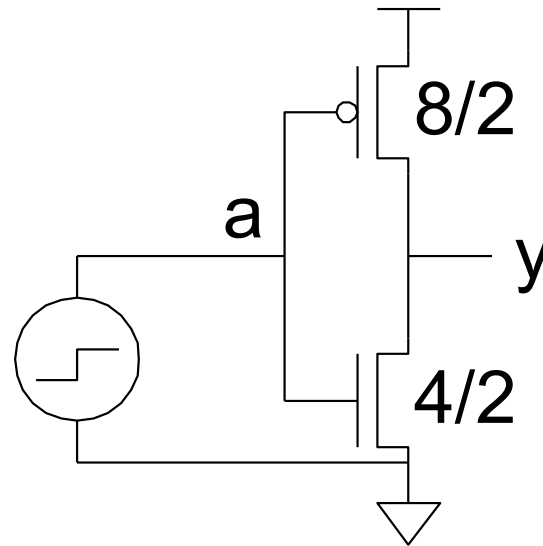


# Inverter Transient Analysis

```

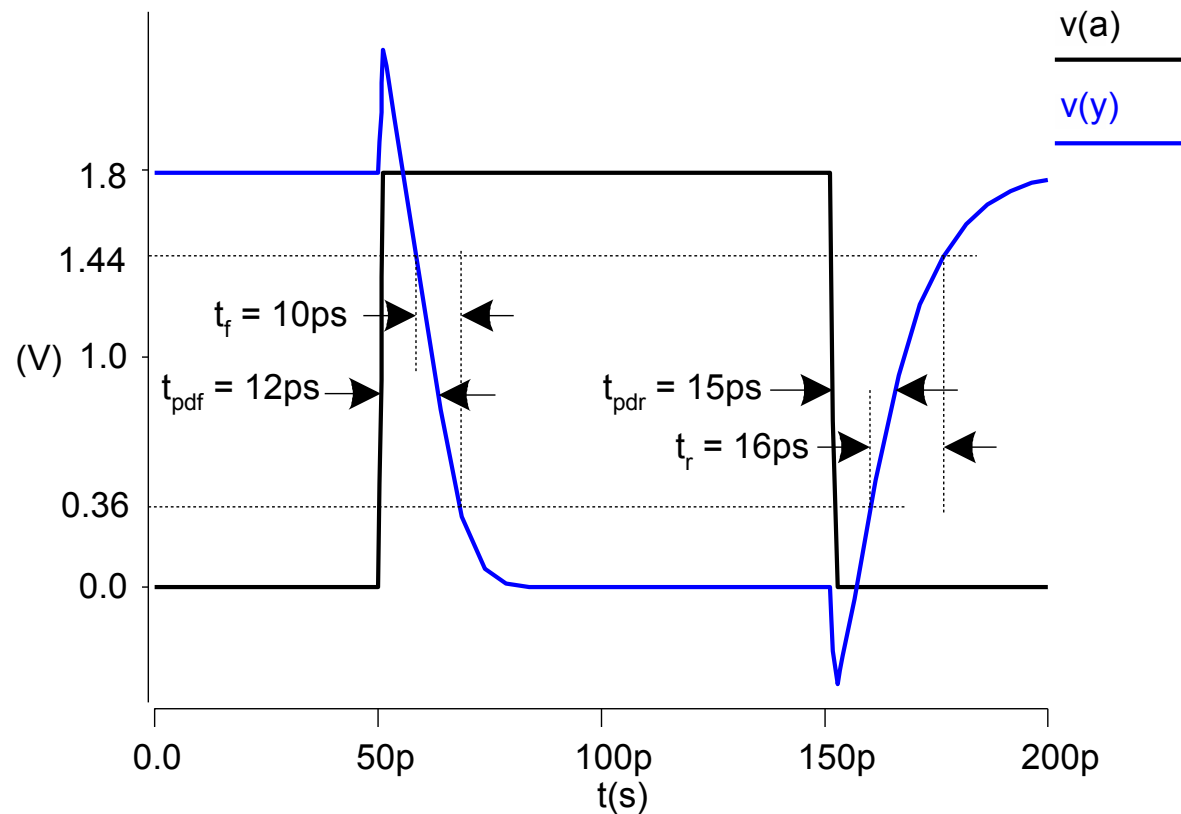
* inv.sp
* Parameters and models
* -----
.param SUPPLY=1.8
.option scale=90n
.include '../models/tsmc180/models.sp'
.temp 70
.option post
* Simulation netlist
* -----
Vdd vdd gnd 'SUPPLY'
Vin a gnd PULSE 0 'SUPPLY' 50ps 0ps 0ps 100ps 200ps
M1 y a gnd gnd NMOS W=4 L=2
+ AS=20 PS=18 AD=20 PD=18
M2 y a vdd vdd PMOS W=8 L=2
+ AS=40 PS=26 AD=40 PD=26
* Stimulus
* -----
.tran 1ps 200ps
.end

```



# Inverter Transient Analysis ....

- Unloaded inverter
  - Overshoot
  - Very fast edges





# Analysis: Pole-Zero

- The pole-zero analysis portion of SPICE computes the poles and/or zeros in the small-signal ac transfer function.
- The program first computes the dc operating point and then determines the linearized, small-signal models for all the nonlinear devices in the circuit.
- Two types of transfer functions are allowed : one of the form (output voltage)/(input voltage) and the other of the form (output voltage)/(input current).



# Analysis: Pole-Zero

- In pole/zero analysis, a network is described by its network transfer function which, for any linear time-invariant network is of the form:

$$H(s) = \frac{N(s)}{D(s)} = \frac{a_0 s^m + a_1 \cdot s^{(m-1)} + \dots + a_m}{b_0 s^n + b_1 \cdot s^{(n-1)} + \dots + b_n}$$

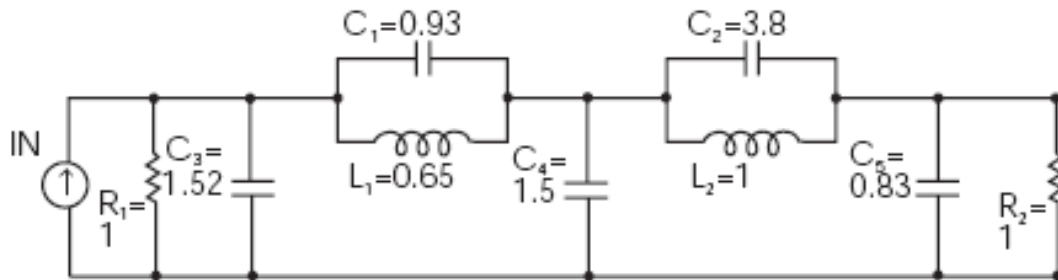
- In the factorized form, the general function is:

$$H(s) = \frac{a_0 \cdot (s + z_1)(s + z_2) \dots (s + z_i) \dots (s + z_m)}{b_0 \cdot (s + p_1)(s + p_2) \dots (s + p_j) \dots (s + p_m)}$$

- The roots of the numerator  $N(s)$  (that is,  $z_i$ ) are called the zeros of the network function, and the roots of the denominator  $D(s)$  (that is,  $p_j$ ) are called the poles of the network function.  $S$  is a complex frequency.
- The dynamic behavior of the network depends upon the location of the poles and zeros on the network function curve. The poles are called the natural frequencies of the network.



# Analysis: Pole-Zero



Low-Pass Prototype Filter

Poles (rad/sec)		Poles (hertz)	
Real	Imag	Real	Imag
-6.948473e-02	-4.671778e-01	-1.105884e-02	-7.435365e-02
-6.948473e-02	4.671778e-01	-1.105884e-02	7.435365e-02

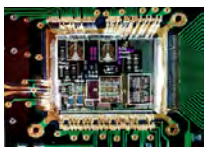
..... and more

## SPICE Deck

```
.OPTIONS POST
.PZ I(R2) IN
.AC DEC 100 .001HZ 10HZ
.PLOT AC IDB(R2) IP(R2)

IN 0 1 1.00 AC 1
R1 1 0 1.0

C3 1 0 1.52
C4 2 0 1.50
C5 3 0 0.83
C1 1 2 0.93
L1 1 2 0.65
C2 2 3 3.80
L2 2 3 1.00
R2 3 0 1.00
.END
```



# Analysis: Small-Signal Distortion

- The distortion analysis portion of SPICE computes steady-state harmonic and inter-modulation products for small input signal magnitudes.
- Distortion analysis is supported for the following nonlinear devices: diodes (DIO), BJT, JFET, MOSFETs (levels 1, 2, 3, 4/BSIM1, 5/BSIM2, and 6) and MESFETs.
- All linear devices are automatically supported by distortion analysis.



# Analysis: Small-Signal Distortion

- The .DISTO statement computes the distortion characteristics of the circuit in an AC small-signal, sinusoidal, steady-state analysis.
- SPICE computes and reports five distortion measures at the specified load resistor. The analysis is performed assuming that one or two signal frequencies are imposed at the input.

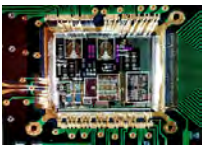
## ➤ Syntax

```
.DISTO Rload <inter <skw2 <refpwr <spwf>>>>
```

- Rload -- the resistor element name,
- inter -- interval at which a distortion-measure summary is to be printed
- Skw2-- ratio of second frequency F2 to nominal analysis frequency F1
- refpwr -- reference power level
- spwf -- amplitude of the second frequency

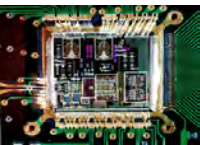
## ➤ Example

```
.DISTO RL 2 0.95 1.0E-3 0.75
```



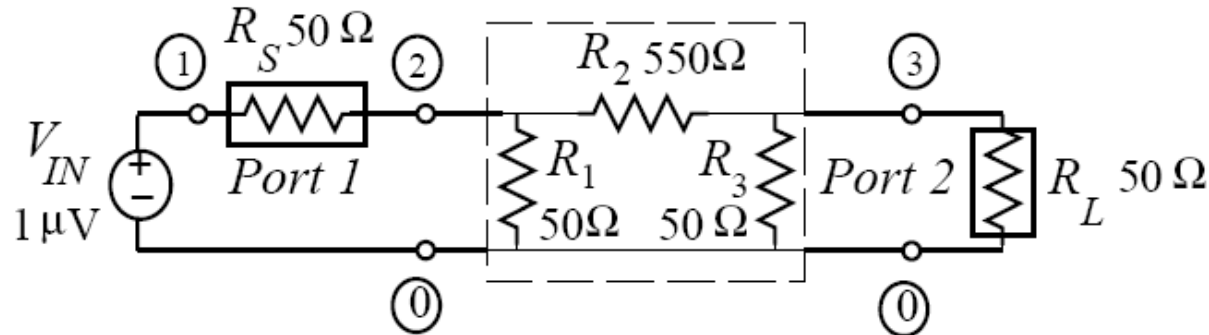
# Noise Analysis

- The noise analysis portion of SPICE does analysis device-generated noise for the given circuit.
- When provided with an input source and an output port, the analysis calculates the noise contributions of each device (and each noise generator within the device) to the output port voltage.
- It also calculates the input noise to the circuit, equivalent to the output noise referred to the specified input source.



# Noise Analysis

2 PORT



```
vin 1 0 AC 1u RS=50 SNR=100
```

For a given input SNR

```
RS 1 2 50
```

```
R1 2 0 55
```

```
R2 2 3 500
```

```
R3 3 0 55
```

```
RL 3 0 50
```

Find output SNR

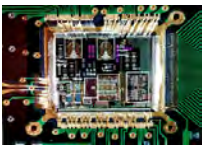
```
.TEMP 16.85
```

```
.AC DEC 1 1MEG 2G
```

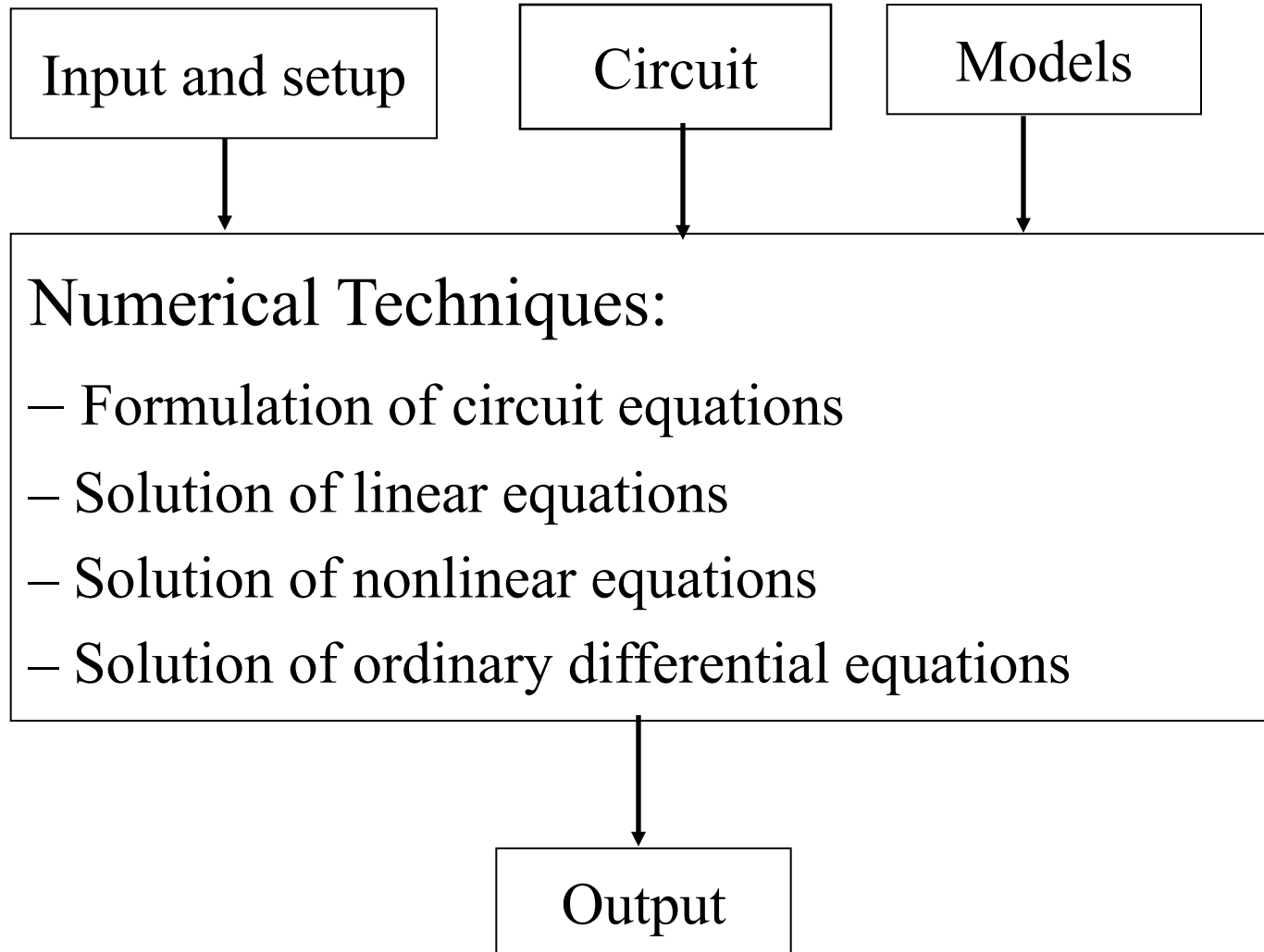
```
.NOISE V(3,0) VIN 1
```

```
.PRINT NOISE nf db(nf) gt db(gt) gain snr inoise onoise
```

```
.END
```

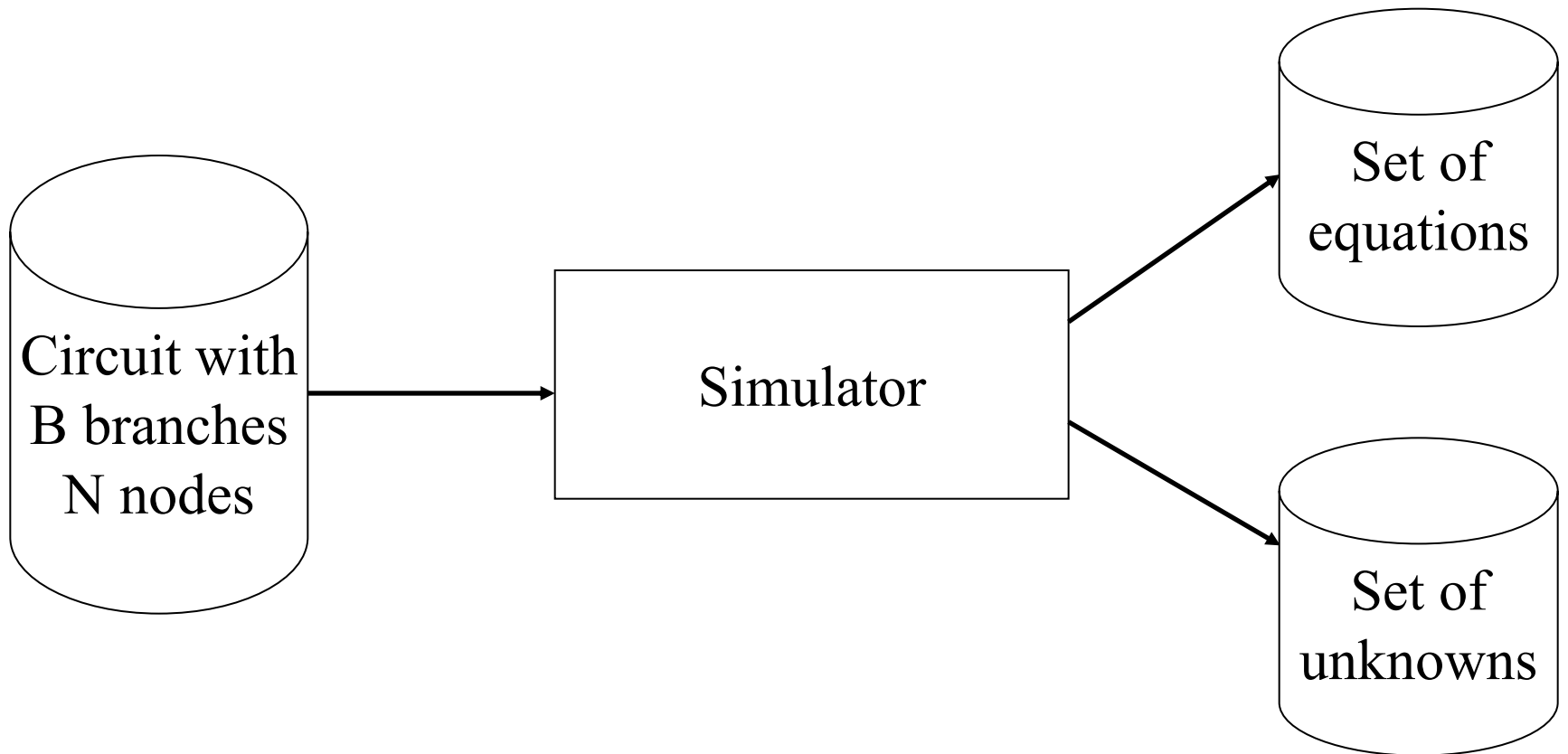


# Program Structure (a closer look)





# Formulation of Circuit Equations



# Formulation of Circuit Equations

- **Unknowns**

- B branch currents (i)
- $4242N$  node voltages (e)
- B branch voltages (v)

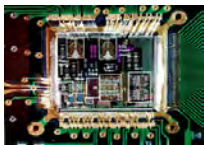
- **Equations**

- $N+B$  Conservation Laws
- B Constitutive Equations



# Conservation Laws

- Determined by the topology of the circuit
  - Kirchhoff's Voltage Law (KVL): Every circuit node has a unique voltage with respect to the reference node. The voltage across a branch  $e_b$  is equal to the difference between the positive and negative referenced voltages of the nodes on which it is incident.
  - Kirchhoff's Current Law (KCL): The algebraic sum of all the currents flowing out of (or into) any circuit node is zero.

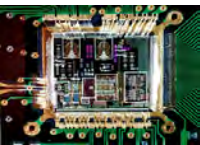


# SPICE: Simplistic View

- Really, it's just all about solving KCL and KVL equations:

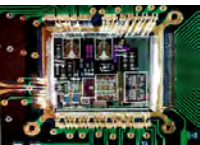
$$\sum I_k = 0, \quad \sum V_k = 0$$

- Generally, it's a nonlinear differential equation (e.g. transistors).
- Bunch of numerical algebra



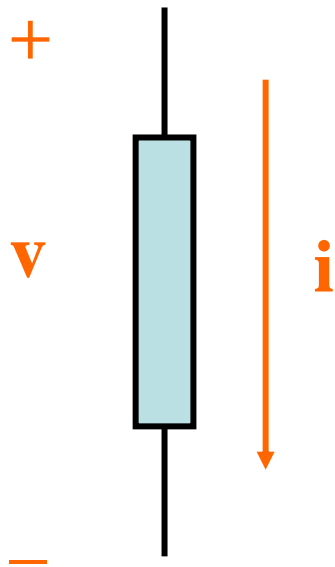
# Branch Constitutive Equations (BCE)

- Determined by the mathematical model of the electrical behavior of a component
  - Example:  $V=R \cdot I$
- In most of circuit simulators this mathematical model is expressed in terms of ideal elements

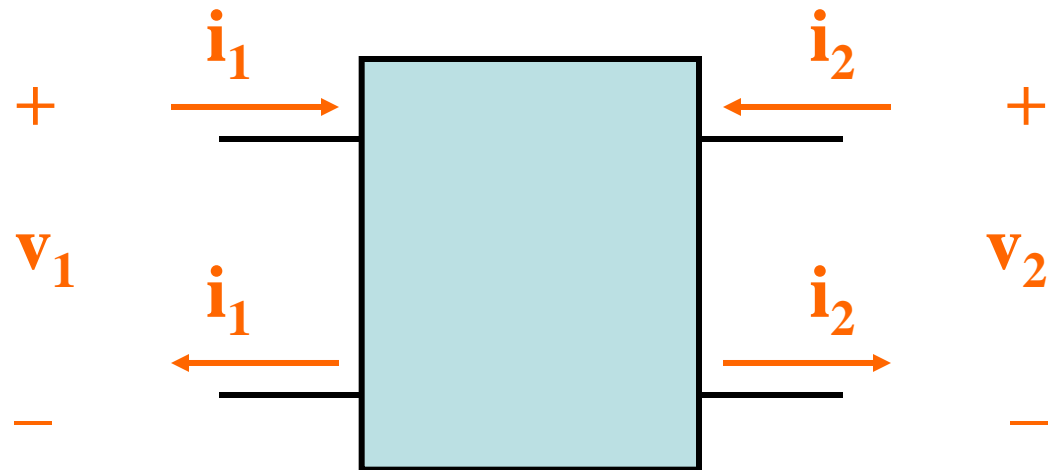


# Ideal Elements: Reference Direction

Two-terminal



Two-port



Branch voltages and currents are measured according to the associated reference directions

–Also define a reference node (ground)



# Branch Constitutive Equations (BCE)

## Ideal elements

Element	Branch Eqn
Resistor	$v = R \cdot i$
Capacitor	$i = C \cdot dv/dt$
Inductor	$v = L \cdot di/dt$
Voltage Source	$v = v_s, i = ?$
Current Source	$i = i_s, v = ?$
VCVS	$v_s = A_V \cdot v_c, i = ?$
VCCS	$i_s = G_T \cdot v_c, v = ?$
CCVS	$v_s = R_T \cdot i_c, i = ?$
CCCS	$i_s = A_I \cdot i_c, v = ?$



# Analysis: AC Small Signal, Theory

We compute the tiny small signal current  $DI$  by using a Taylor series expansion about the bias point i.e.

$$I = f(v)$$

so

$$I_{dc} + DI = f(V_{dc}) + f'(V_{dc})DV$$



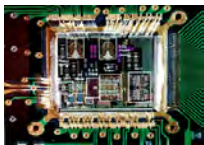


# Analysis: AC Small Signal, Theory ...

- The problem is split into two parts
- A nonlinear dc part where there is no time variation,  $I_{dc} = f(V_{dc})$
- A linear time-varying part

$$DI = f'(V_{dc})DV$$

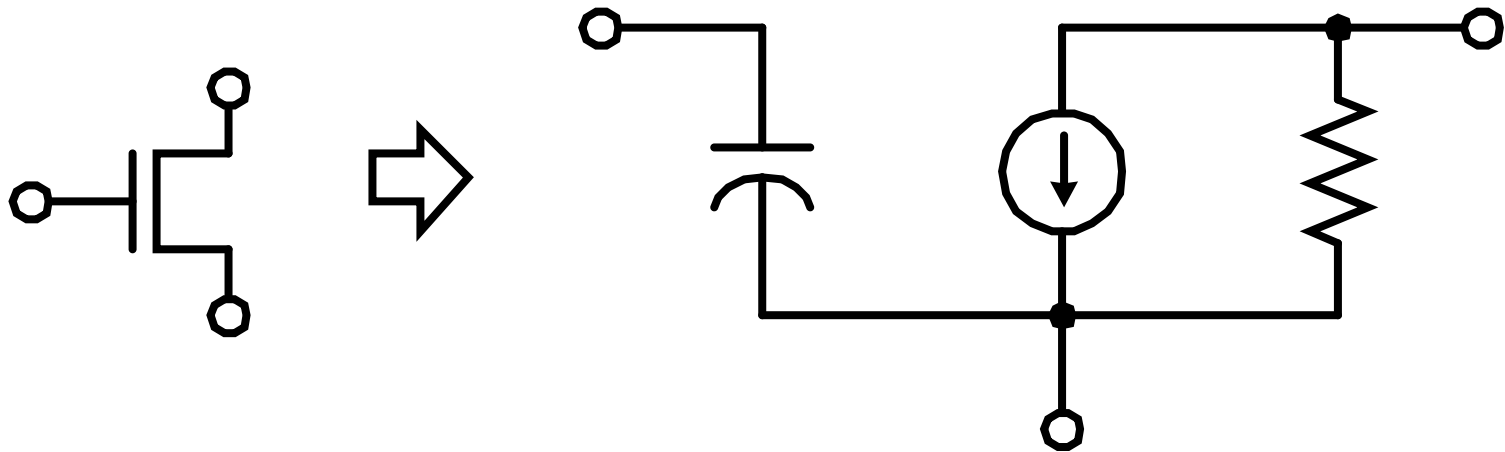
- The derivative is represented by a conductance or resistance in a small-signal equivalent circuit.



# Analysis: AC Small Signal, Theory ...

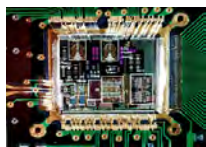
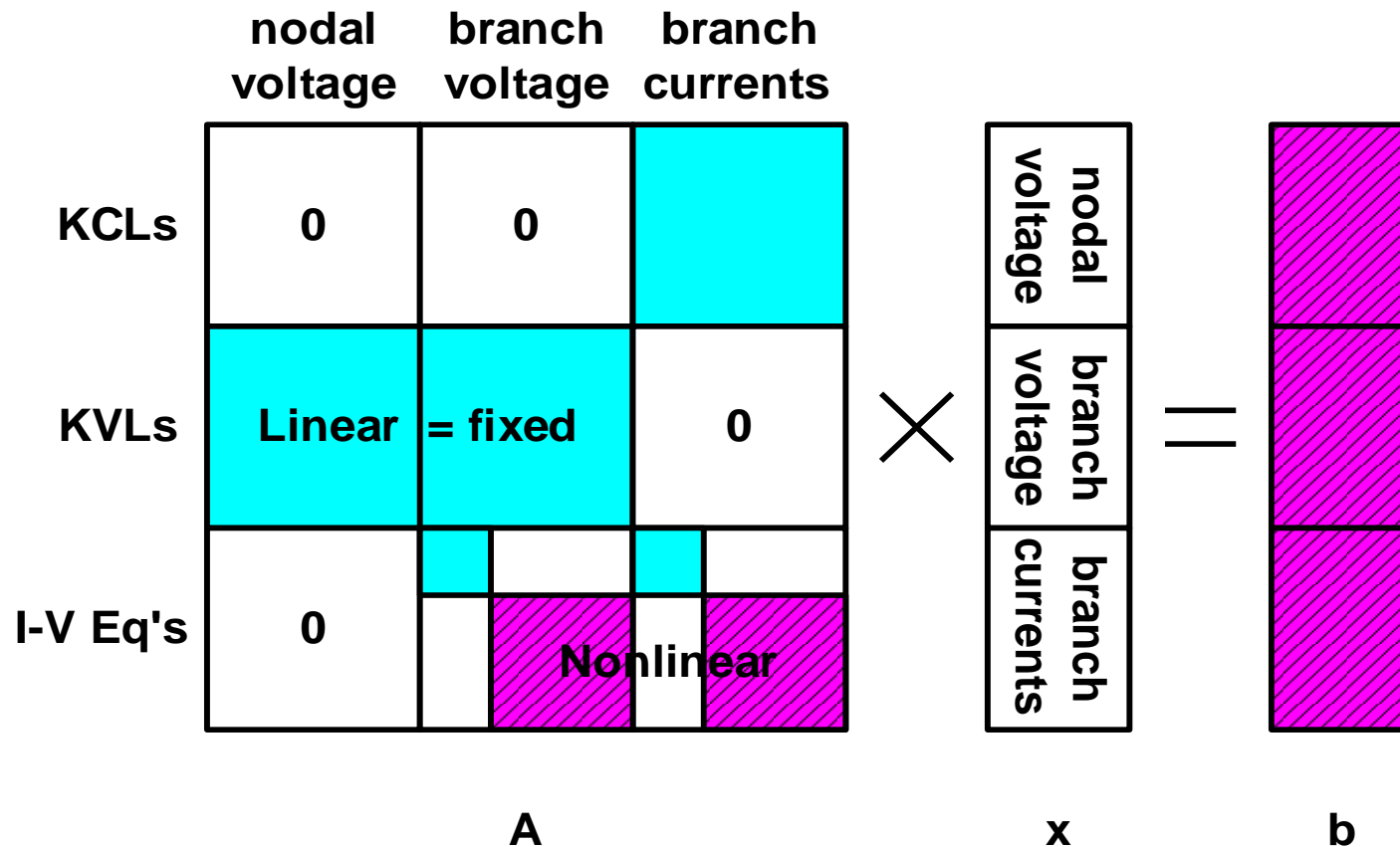
## First, Linearize it

At each time step, SPICE builds a small-signal model (i.e. linear model) at the operating point:



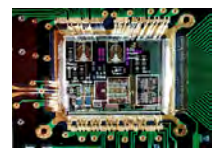
# Matrix Equation

Construct a matrix equation :  $Ax=b$



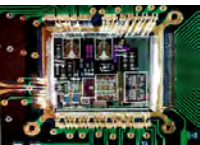
# Methods for Linear & Non-Linear Systems

- Systems of linear equations
  - Existence and uniqueness review
  - Gaussian Elimination basics
    - LU factorization
    - Pivoting
- Nonlinear problems
- Iterative Methods
- Newton's Method
  - Derivation of Newton
  - Quadratic Convergence
  - Examples



# Systems of Linear Equations

- Problem to solve:  $Mx = b$
- Given  $Mx = b$  :
  - Is there a solution?
  - Is the solution unique?



# Systems of Linear Equations ...

$$\begin{bmatrix} \uparrow & \uparrow & \dots & \uparrow \\ \vec{M}_1 & \vec{M}_2 & \dots & \vec{M}_N \\ \downarrow & \downarrow & \dots & \downarrow \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}$$



$$x_1 \vec{M}_1 + x_2 \vec{M}_2 + \dots + x_N \vec{M}_N = b$$

Find a set of weights  $x$  so that the weighted sum of the columns of the matrix  $M$  is equal to the right hand side  $b$



# Systems of Linear Equations - Existence

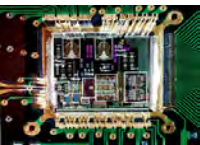
A solution exists if:

There exist weights,  $x_1, \dots, x_N$ , such that:

$$x_1 \vec{M}_1 + x_2 \vec{M}_2 + \dots + x_N \vec{M}_N = b$$



A solution exists when  $b$  is in the span of the columns of  $M$



# Systems of Linear Equations - Uniqueness

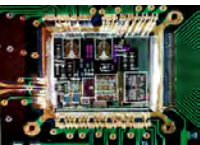
Suppose there exist weights,  $y_1, \dots, y_N$ , not all zero, such that:

$$y_1 \vec{M}_1 + y_2 \vec{M}_2 + \dots + y_N \vec{M}_N = \mathbf{0}$$

Then:  $Mx = b \rightarrow Mx + My = b \rightarrow M(x+y) = b$



A solution is unique only if the columns of  $M$  are linearly independent.





# Methods for Solving Linear Equations

- **Direct methods:** find the exact solution in a finite number of steps
- **Iterative methods:** produce a sequence a sequence of approximate solutions hopefully converging to the exact solution



# Gaussian Elimination Basics

Gaussian Elimination Method for Solving  $Mx = b$

- A “Direct” Method Finite Termination for exact result (ignoring round-off)
- Produces accurate results for a broad range of matrices
- Computationally Expensive



# Gaussian Elimination (GE) Basics

Reminder by 3x3 example

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$



$$M_{11}x_1 + M_{12}x_2 + M_{13}x_3 = b_1$$

$$M_{21}x_1 + M_{22}x_2 + M_{23}x_3 = b_2$$

$$M_{31}x_1 + M_{32}x_2 + M_{33}x_3 = b_3$$



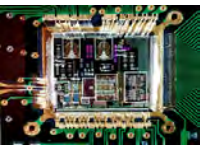
# GE Basics – Key idea in the matrix

Remove  $x_1$  from eqn 2 and eqn 3

Pivot

MULTIPLIERS

$$\begin{bmatrix}
 M_{11} \\
 0 \\
 0
 \end{bmatrix}
 \begin{bmatrix}
 M_{12} \\
 \left( M_{22} - \left( \frac{M_{21}}{M_{11}} \right) M_{12} \right) \\
 \left( M_{32} - \left( \frac{M_{31}}{M_{11}} \right) M_{12} \right)
 \end{bmatrix}
 \begin{bmatrix}
 M_{13} \\
 \left( M_{23} - \left( \frac{M_{21}}{M_{11}} \right) M_{12} \right) \\
 \left( M_{33} - \left( \frac{M_{31}}{M_{11}} \right) M_{12} \right)
 \end{bmatrix}
 \begin{bmatrix}
 x_1 \\
 x_2 \\
 x_3
 \end{bmatrix}
 =
 \begin{bmatrix}
 b_1 \\
 b_2 - \left( \frac{M_{21}}{M_{11}} \right) b_1 \\
 b_3 - \left( \frac{M_{31}}{M_{11}} \right) b_1
 \end{bmatrix}$$

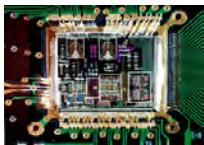


# GE Basics – Simplify the Notation

Remove  $x_2$  from eqn 3

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ 0 & \tilde{M}_{22} & \tilde{M}_{23} \\ 0 & 0 & \tilde{M}_{33} - \frac{\tilde{M}_{32} \tilde{M}_{23}}{\tilde{M}_{22}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ \tilde{b}_2 \\ \tilde{b}_3 - \frac{\tilde{M}_{32} \tilde{b}_2}{\tilde{M}_{22}} \end{bmatrix}$$

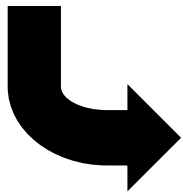
Pivot (points to  $\tilde{M}_{22}$ )  
Multiplier (points to  $\frac{\tilde{M}_{32} \tilde{M}_{23}}{\tilde{M}_{22}}$ )



# GE Basics – Yielding Triangular System

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ 0 & \tilde{M}_{22} & \tilde{M}_{23} \\ 0 & 0 & \tilde{\tilde{M}}_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ \tilde{b}_2 \\ \tilde{\tilde{b}}_3 \end{bmatrix}$$

Altered During GE

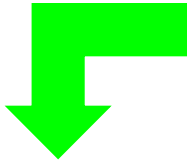


$$\begin{bmatrix} U_{11} & U_{12} & U_{13} \\ 0 & U_{22} & U_{23} \\ 0 & 0 & U_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$



# GE Basics – Backward Substitution

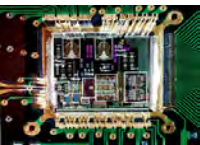
$$\begin{bmatrix} U_{11} & U_{12} & U_{13} \\ 0 & U_{22} & U_{23} \\ 0 & 0 & U_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$



$$x_3 = \frac{y_3}{U_{33}}$$

$$x_2 = \frac{y_2 - U_{23}x_3}{U_{22}}$$

$$x_1 = \frac{y_1 - U_{12}x_2 - U_{13}x_3}{U_{11}}$$



# GE Basics: Summary

(1)  $Mx = b$   
GE  
 $Ux = y$

Equivalent system

$U$ : upper trg

(2) Noticed that:

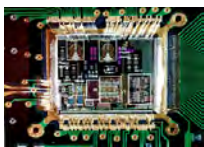
$$Ly = b$$

$L$ : unit lower trg

(3)  $Ux = y$

$$LUx = b \rightarrow Mx = b$$

$\Rightarrow$  Efficient way of implementing GE: LU factorization





# Gaussian Elimination Basics

Solve  $M x = b$

Step 1

$$M = L U$$

$$\square = \triangle \circ$$

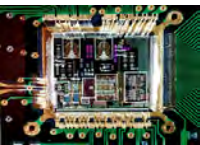
Step 2 Forward Elimination

Solve  $L y = b$

Step 3 Backward Substitution

Solve  $U x = y$

Note: Changing RHS does not imply to recompute LU factorization

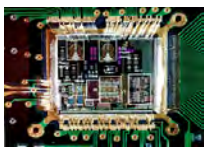


# GE Basics – Fitting the Pieces Together

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}$$

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{M_{21}}{M_{11}} & 1 & 0 \\ \frac{M_{31}}{M_{11}} & \frac{\tilde{M}_{32}}{\tilde{M}_{22}} & 1 \\ \frac{M_{31}}{M_{11}} & \tilde{M}_{22} & 1 \end{bmatrix}$$

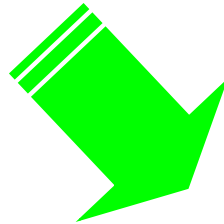
$$U = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ 0 & \tilde{M}_{22} & \tilde{M}_{23} \\ 0 & 0 & \tilde{M}_{33} \end{bmatrix}$$



# GE Basics – Fitting the Pieces Together

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{M_{21}}{M_{11}} & 1 & 0 \\ \frac{M_{31}}{M_{11}} & \frac{\tilde{M}_{32}}{\tilde{M}_{22}} & 1 \\ \frac{M_{11}}{M_{11}} & \frac{\tilde{M}_{22}}{\tilde{M}_{22}} & 0 \end{bmatrix}$$

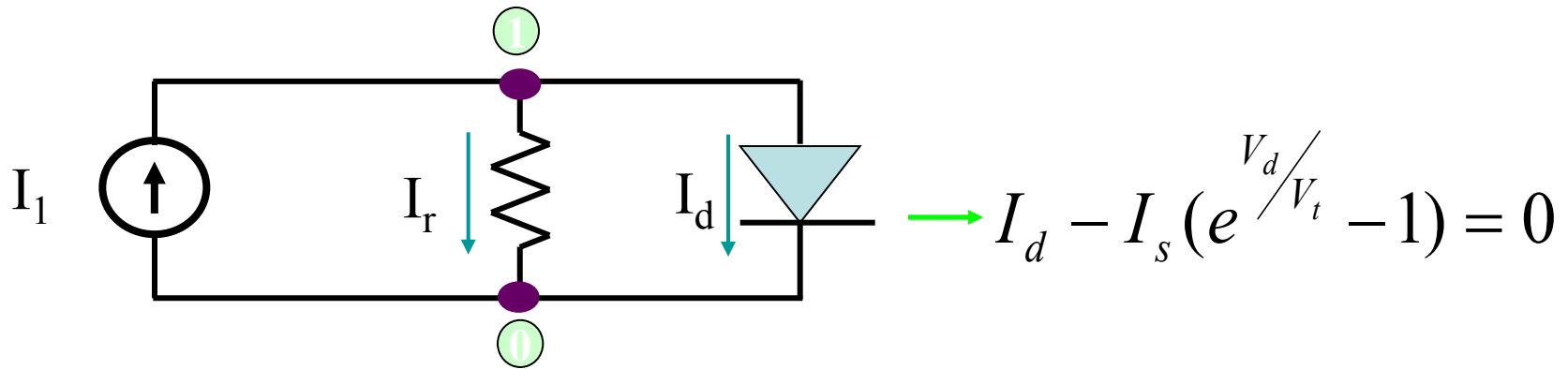
$$U = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ 0 & \tilde{M}_{22} & \tilde{M}_{23} \\ 0 & 0 & \tilde{M}_{33} \end{bmatrix}$$



$$\begin{bmatrix} M_{11} & M_{21} & M_{31} \\ \frac{M_{21}}{M_{11}} & \tilde{M}_{22} & \tilde{M}_{23} \\ \frac{M_{31}}{M_{11}} & \frac{\tilde{M}_{32}}{\tilde{M}_{22}} & \tilde{M}_{33} \\ \frac{M_{11}}{M_{11}} & \frac{\tilde{M}_{22}}{\tilde{M}_{22}} & 0 \end{bmatrix}$$



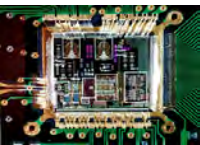
# Nonlinear Problems - Example



Need to Solve

$$\boxed{I_r + I_d - I_1 = 0}$$

$$\frac{1}{R} e_1 + I_s (e^{\frac{e_1}{V_t}} - 1) - I_1 = 0 \quad \longrightarrow \quad g(e_1) = I_1$$



# Nonlinear Equations

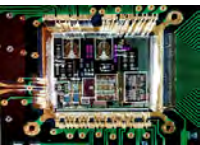
- Given  $g(V)=I$

- It can be expressed as:  $f(V)=g(V)-I$

$\Rightarrow$  Solve  $g(V)=I$  equivalent to solve  $f(V)=0$

Hard to find analytical solution for  $f(x)=0$

Solve iteratively



# Nonlinear Equations – Iterative Methods

- Start from an initial value  $x^0$
- Generate a sequence of iterate  $x^{n-1}, x^n, x^{n+1}$  which hopefully converges to the solution  $x^*$
- Iterates are generated according to an iteration function  $F: x^{n+1} = F(x^n)$

Questions:

- When does it converge to correct solution ?
- What is the convergence rate ?



# Newton-Raphson (NR) Method

Consists of linearizing the system.

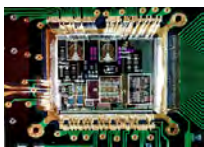
Want to solve  $f(x)=0 \rightarrow$  Replace  $f(x)$  with its linearized version and solve.

$$f(x) = f(x^*) + \frac{df}{dx}(x^*)(x - x^*) \quad \textit{Taylor Series}$$

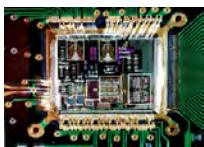
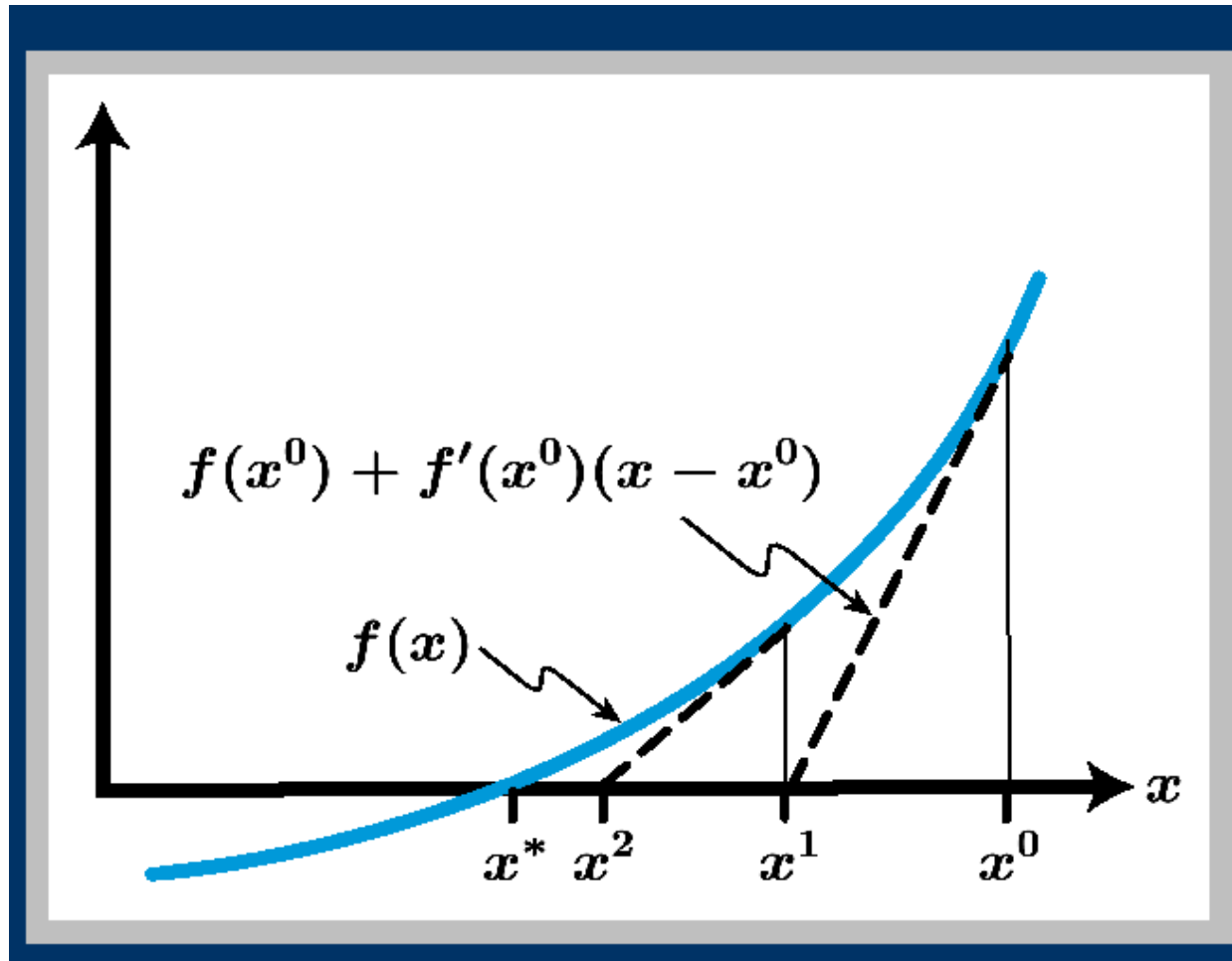
$$f(x^{k+1}) = f(x^k) + \frac{df}{dx}(x^k)(x^{k+1} - x^k)$$

$$\Rightarrow x^{k+1} = x^k - \left[ \frac{df}{dx}(x^k) \right]^{-1} f(x^k) \quad \textit{Iteration function}$$

Note: at each step need to evaluate  $f$  and  $f'$



# Newton-Raphson Method – Graphical





# Newton-Raphson Method – Algorithm

Define iteration

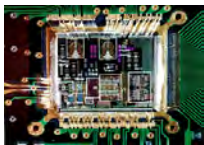
Do  $k = 0$  to .....

$$x^{k+1} = x^k - \left[ \frac{df}{dx} (x^k) \right]^{-1} f(x^k)$$

until convergence

- How about convergence?
- An iteration  $\{x^{(k)}\}$  is said to converge with order  $q$  if there exists a vector norm such that for each  $k \geq N$ :

$$\|x^{k+1} - \hat{x}\| \leq \|x^k - \hat{x}\|^q$$



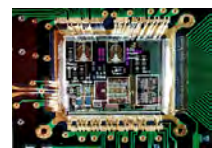
# Newton-Raphson Method – Convergence

## Local Convergence Theorem

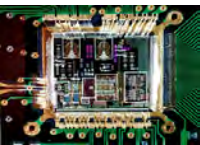
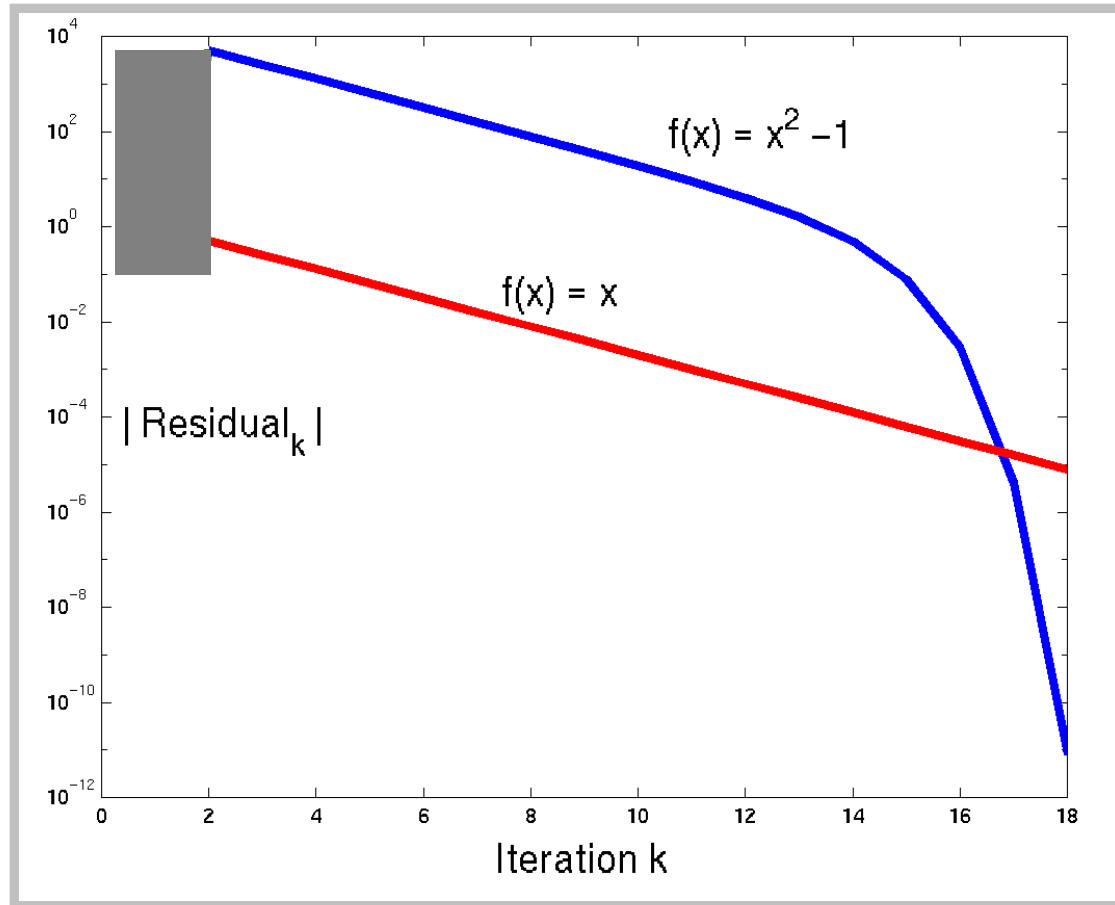
If

$$\left. \begin{array}{l} a) \frac{df}{dx} \quad \textit{bounded away from zero} \\ b) \frac{d^2 f}{dx^2} \quad \textit{bounded} \end{array} \right\} K \textit{ is bounded}$$

Then Newton's method converges given a sufficiently close initial guess (and convergence is quadratic)



# Newton-Raphson Method – Convergence



# Reference

- [http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/UserGuide/overview\\_fr.html](http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/UserGuide/overview_fr.html)
- [http://bear.ces.cwru.edu/eecs\\_cad/cad\\_spice.html](http://bear.ces.cwru.edu/eecs_cad/cad_spice.html)
- [http://mos.stanford.edu/papers/jk\\_fspice.ppt](http://mos.stanford.edu/papers/jk_fspice.ppt)
- <http://www.ecircuitcenter.com/AboutSPICE.htm>
- <http://www-cad.eecs.berkeley.edu/~nardi/EE219A/lectures/>
- <http://www.ieee.uta.edu/Tutorials/pspice/Basic%20Tuto.ppt>

