Simulation Using WinSPICE

David W. Graham

Lane Department of Computer Science and Electrical Engineering

West Virginia University

Why Simulation?

- Theoretical calculations only go so far…
- Find out the circuit behavior in a variety of operating conditions
- It is currently the best way of designing a circuit (industry standard)
- Provides intuitive “feel” for circuit operation (without requiring expensive equipment)

Simulator Options

Wide variety of circuit simulators

- Specialized simulators (typically discrete-time)
  - Multitude of digital simulators
  - Switcap (for switched-capacitor circuits)
- Generic simulators (analog / continuous-time circuits typically use these)
  - SPICE (Simulation Program with Integrated Circuit Emphasis)

SPICE Options Available at WVU

- HSPICE
  - Good
  - Expensive
  - Different syntax
- PSPICE
  - Schematic capture
  - Node limitation (9 nodes maximum)
- WinSPICE
  - Free! (Plus, it is good in many other ways)

WinSPICE

Pros

- Free
- Small Size
- Can run it from MATLAB
- Works well
- No node limitations
- Can use the EKV model (good for subthreshold simulations)
- Works in Windows

Cons

- No schematic capture (Rumor – XCircuit can perform schematic capture)
- Only works in Windows
- (Occasional convergence problems – but improving)

How to Obtain WinSPICE

- Free download
- www.winspice.com
- Go to “Download”
  - Download “Current Full Version”
  - Then, download the current stable release (this is simply an update)

Writing SPICE Decks / Netlists

SPICE Deck/Netlist is a text description of a circuit

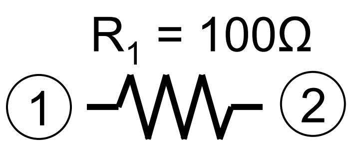
Consists of the following parts

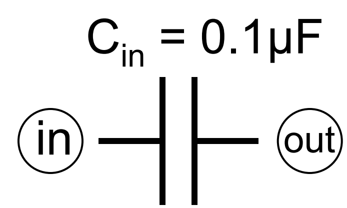
- Header
- Circuit connections
- Subcircuit descriptions (if needed)
- Model descriptions (if needed – usually only for transistors)
- Analyses to be performed
- Outputs to be saved / displayed

Basic Circuit Elements

- Resistor `R<label> node1 node2 value`
- Capacitor `C<label> node1 node2 value`
- Inductor `L<label> node1 node2 value`

Examples






```
R1 1 2 100
```

Resistor “name”

Signifies resistor

```
CIN IN OUT 0.1u
```

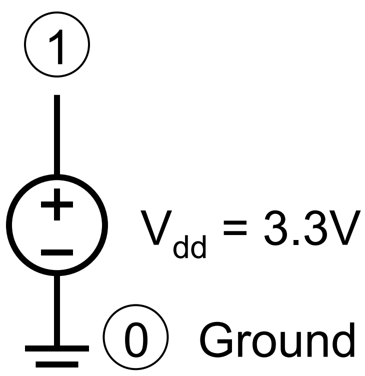
Signifies “micro” (1e-6)

Nodes can be signified by words instead of numbers

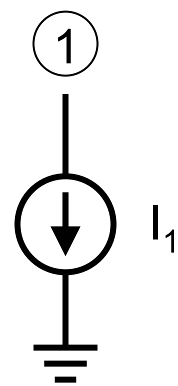
Independent Voltage and Current Sources

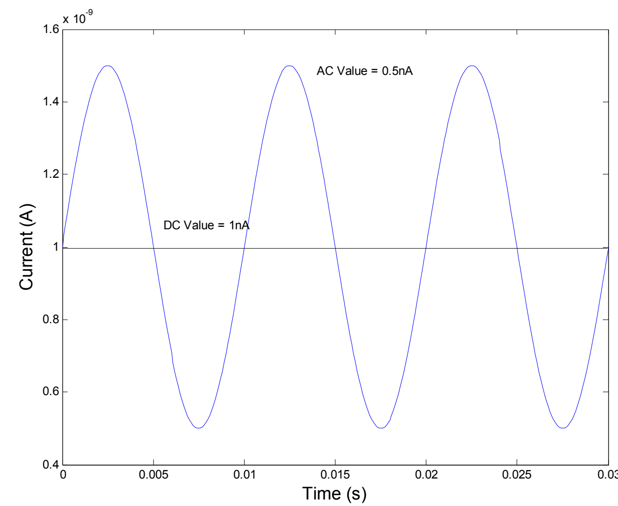
- Voltage Source V<name> n+ n- DC dcvalue AC acvalue
- Current Source I<name> n+ n- DC dcvalue AC acvalue

Examples



Ground is <u>always</u> node 0






```
VDD 1 0 DC 3.3 AC 0
```

Direction of current flow

```
I1 1 0 DC 1n AC 0.5e-9
```

n = 1e-9 (equivalent forms)

Independent Voltage and Current Sources

Independent sources can also output functions

- PULSE – Pulse function
- PWL – Piecewise linear function
- SIN – Sinusoidal waveform
- EXP – Exponential waveform
- SFFM – Single-frequency FM

For more information, see the SPICE manual (WinSPICE manual)

Example – Sinusoidal voltage with a DC offset of 1V, an amplitude of 0.5V, and a frequency of 1kHz (between nodes 1 and 0)

```
V<name> n+ n- SIN(dcvalue amplitude frequency)
V1 1 0 SIN(1 0.5 1k)
```

Dependent Voltage and Current Sources

- Voltage-controlled voltage source (VCVS)

  `E<label> n+ n- nref+ nref- gain`
- Current-controlled current source (CCCS)

  `F<label> n+ n- voltagesourceref gain`
- Voltage-controlled current source (VCCS)

  `G<label> n+ n- nref+ nref- transconductance`
- Current-controlled voltage source (CCVS)

  `H<label> n+ n- voltagesourceref transconductance`

- Voltage-controlled sources reference the voltage across two nodes
- Current-controlled sources reference the current flowing through a voltage source
  - Can be a “dummy” voltage source
  - A voltage source with no voltage supplied
  - `VDUMMY 3 4 DC 0 AC 0`
- Current sources flow from n+ to n-

Transistors

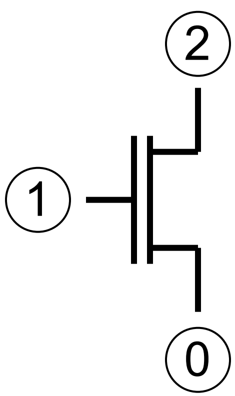
- nFETs

```
M<name> drain gate source bulk modelname W=value L=value
```

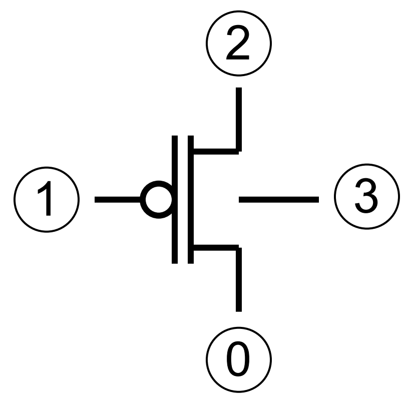
- pFETs

```
M<name> drain gate source well modelname W=value L=value
```

Examples (Assume models “NFET” and “PFET” are defined elsewhere)



Assume the bulk connection is tied to ground




```
M1 2 1 0 0 NFET W=100u L=4.8u
```

```
M2 0 1 2 3 PFET W=100u L=4.8u
```

Model Files

Two major models for simulating transistors

- BSIM
  - Great for above threshold simulations
  - Essentially empirical fits
  - Many, many parameters (upwards of hundreds)
  - Does <u>not</u> do subthreshold very well, at all
- EKV Model
  - Mathematical model of the MOSFET operation
  - Much fewer parameters
  - Does subthreshold operation very well

EKV Model

- Enz, Krummenacher, and Vittoz Model
  - (3 Swiss engineers who wanted a better MOSFET model, specifically for low-current applications)
- Model is a “single expression” that preserves continuity of the operation
- Based on the physics of the MOS device (not just empirical fits)

- We will be using the 0.5μm model available at the EKV website
  - http://legwww.epfl.ch/ekv/ekv26_0u5.par
- More information can be found at
  - http://legwww.epfl.ch/ekv/
  - Liu, et al. pg 86-89

Analysis

Several types of analyses can be performed

- Operating point
- DC sweep
- AC sweep
- Transient analysis

We will be making use of these analyses extensively

- Additional useful analyses – distortion, noise, pole-zero, sensitivity, temperature, transfer function

Analysis

Analysis declaration is given by a line of code near the end of the SPICE deck

- Operating point analysis (.OP)
  - Provides DC operating point (capacitors shorted, inductors opened)
  - `.OP`
- DC sweep (.DC)
  - Can sweep a DC voltage or current to determine a DC transfer function
  - `.DC sourcename startval stopval incrementval`
  - e.g. → `.DC VIN 0 5 0.1` (This would sweep source VIN from 0V to 5V with steps of 0.1V)

Analysis

- AC analysis (.AC)
  - Can sweep an AC voltage or current over a specified frequency range to determine the transfer function / frequency response
  - Does <u>not</u> take distortion and nonlinearities into account
  - `.AC {DEC,OCT,LIN} numpoints freqstart freqstop`
    - `DEC` – `numpoints` per decade
    - `OCT` – `numpoints` per octave
    - `LIN` – linear spacing of points, `numpoints` = total number of points
  - e.g. → .AC DEC 10 10 1E5
    - AC sweep from 10Hz to 100kHz, points spaced logarithmically, 10 simulation points per decade
  - Must have a source with an AC component in the circuit

Analysis

- Transient analysis (.TRAN)
  - Determines the response of a circuit to a transient signal / source (sine wave, PWL function, etc.)
  - Allows you to achieve the most results with a simulation (distortion, nonlinearity, operation, etc.)
  - `.TRAN timestep timestop {timestart {maxstepsize}} {UIC}`
  - Optional arguments
    - timestart = start time (default is 0)
    - maxstepsize = maximum time increment between simulation points
    - UIC – "Use Initial Conditions" – allows the user to define initial conditions for start of simulation, e.g. initial voltage on a capacitor
  - e.g. → `.TRAN 1n 100n`
    - Perform a transient analysis for 100nsec (100e-9 seconds) with a step increment of 1nsec

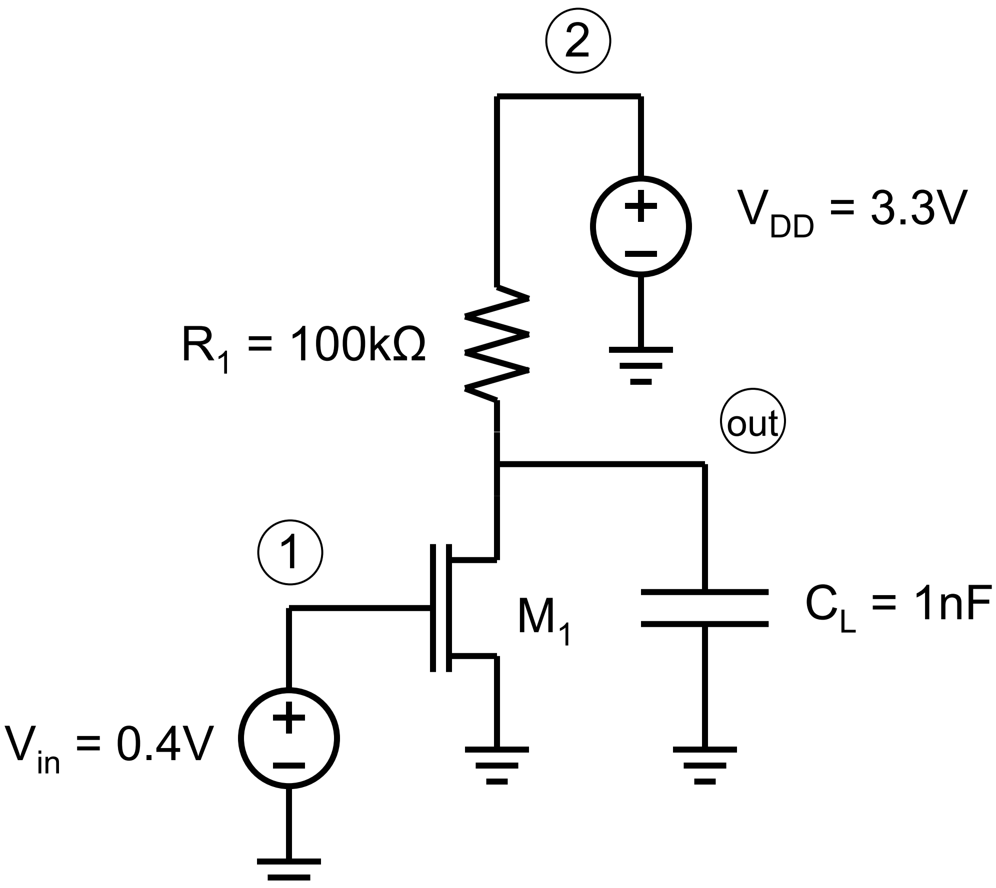
Displaying Outputs

- Saving variables
  - Saving the values of the voltages / currents for use in later plotting them
  - `.SAVE variable1 variable2` …
  - Examples
    - `.SAVE V(1)` (Saves the voltage at node 1)
    - `.SAVE VIN VOUT @M1[ID]` (Saves the voltages at nodes VIN and VOUT, also saves the drain current through transistor M1)
    - `.SAVE ALL` (Saves all variables)

Displaying Outputs

- Plotting variables
  - Plot type depends on the analysis performed
  - `.PLOT analysistype variable1 variable2 …`
  - Examples
    - `.PLOT DC V(1) V(2)` (Plots the voltages at nodes 1 and 2 on the same graph. The x axis is voltage (DC sweep))
    - `.PLOT AC VDB(3)` (Plots the decibel value of the voltage at node 3. The x axis is frequency (AC analysis))
    - `.PLOT TRAN I(VIN)` (Plots the current through the voltage source VIN. The x axis is time (transient analysis))

A Circuit Example




```
COMMON SOURCE AMPLIFIER

*BEGIN CIRCUIT DESCRIPTION
VIN 1 0 DC 1 AC 0
VDD 2 0 DC 3.3 AC 0
R1 OUT 2 100K
CL OUT 0 1N
M1 OUT 1 0 0 NFET L=10U
W=100U
```

<Insert Model Statements Here>

```
.OP
.DC VIN 0 3.3 0.01
.PLOT DC V(OUT)

.END
```

A Circuit Example

Header – First line is always a title / comment

```
COMMON SOURCE AMPLIFIER
```

`*` Comments out the entire line

```
*BEGIN CIRCUIT DESCRIPTION
VIN 1 0 DC 1 AC 0
VDD 2 0 DC 3.3 AC 0
R1 OUT 2 100K
CL OUT 0 1N
M1 OUT 1 0 0 NFET L=10U
W=100U
```

<Insert Model Statements Here>

Analyses and outputs to be displayed

```
.OP
.DC VIN 0 3.3 0.01
.PLOT DC V(OUT)
```

Must end with a `.END` command

```
.END
```

Running a Simulation

- Save your SPICE Deck as a .cir file
- Simply double-click on the file – WinSPICE will automatically run
- As long as WinSPICE is open, every time you save the .cir file, WinSPICE will automatically re-simulate

Controlling Simulations with MATLAB

One nice feature of WinSPICE is that it can be controlled from MATLAB. This allows post-processing of the simulation results to be done in the easy-to-use MATLAB environment.

- Download the MATLAB .m file from the class website named `runwinspice.m`
- Place a copy of the WinSPICE executable file (.exe file) in the same directory as your .cir file
- Make sure you save the variables you want to view with the `.SAVE` command (the fewer variables you save, the faster the simulation runs)
- Comment out / remove all lines that display outputs (plots) in the .cir file
- Run the simulation from MATLAB using
  - `[data, names] = runwinspice('mycircuit.cir');`
  - `data`
    - Matrix of all variables that were saved with the `.SAVE` line
    - Each variable is saved as a column
    - In AC analyses, two columns are required for each variable
      - Odd-numbered columns are the real part of the simulation data
      - Even-numbered columns are the imaginary part of the simulation data
  - `names`
    - List of the names of the variables corresponding to each column in “data”
    - In AC analyses, there are half as many “names” as there are columns in “data”

Advanced Features in SPICE

- Subcircuits (for reusable circuit elements)
- Global lines
- “Include” statements
- Many, many more (see the SPICE manual)

Subcircuits

- Creates a reusable circuit so you do not have to unnecessarily write identical lines of code over and over again
- Has external nodes (for connections)
- Has internal nodes (for the operation of the subcircuit)
- Usage

```
.SUBCKT subcktname extnode1 extnode2 …
```

<Internal circuit connections>

```
.ENDS subcktname
```

- Connection to the circuit (Subcircuit calls)

```
X<label> node1 node2 … subcktname
```

Subcircuit Example

- Define a subcircuit with the following lines of code

```
.SUBCKT INV 1 2
M1 2 1 3 3 PFET W=1.5 L=1.5U
M2 2 1 0 0 NFET W=1.5 L=1.5U
VSUPPY 3 0 DC 3.3 AC 0
.ENDS INV
```

- Call the subcircuit `INV` in the circuit declaration part of the SPICE deck using the following line

```
X1 8 9 INV
```

Declares this subcircuit will be `INV`

Nodes to connect to in the overall circuit

Subcircuit label “1”

Declares this will be a subcircuit

Global Lines

- Global nodes are valid in all levels of the circuit, including the subcircuits
- Especially useful for power supplies (V_{DD})
- Usage

```
.GLOBAL node1 node2 …
```

Include Statements

- Useful for adding large, reusable lines of code
  - Model files
  - Subcircuits
  - Large, specific input signals (PWL)
- Usage
  - `.INCLUDE filename`
  - Effectively replaces the `.INCLUDE` line with the lines of code in the file