SPICE Simulator Tutorial

Advanced Analog IC Design - Graduate Level

ECE 6XX - Advanced Analog IC Design

Graduate Course

June 4, 2025

Focus: Comprehensive SPICE Simulation Types & Operating Modes
Target: Graduate Students with Advanced Circuit Design Background

Outline

- Introduction to SPICE
  - Historical Context & Versions
  - Core Algorithm
- SPICE Input File Architecture
  - Line Types
  - Critical Rules
  - Token Separation
  - Advanced Considerations
  - Common Pitfalls
- Netlist Construction
  - Node Labeling Requirements
  - Circuit Element Types
  - MOSFET Modeling
  - Parasitic Elements
- Independent Source Specifications
  - Multi-Analysis Source Definition
  - DC Analysis Sources
  - AC Analysis Sources
  - Pulse Function

Introduction to SPICE

What is SPICE?

SPICE: Simulation Program with Integrated Circuit Emphasis

- **Origin:** University of California, Berkeley
- **Major Versions:** SPICE2G6 and SPICE3F4
- **Commercial Variants:** PSpice, HSPICE, Spectre, etc.
- **Open Source:** Ngspice, WinSpice, etc.

Modified Nodal Analysis (MNA)

Matrix-based circuit solving method that forms the mathematical foundation for all SPICE variants

Why This Matters for Advanced IC Design

- Industry standard for analog circuit verification
- Essential for understanding simulation limitations and accuracy
- Critical for advanced topics like noise, mismatch, and process variation analysis
- Foundation for automated design flows and optimization

Advanced Applications

- Statistical yield analysis
- Process corner verification
- Parasitic extraction validation
- Design optimization loops

SPICE Input File Architecture

- **Title Line:** First line (any text)
- **Comments:** * in column 1
- **Netlist:** Start with letter
- **Control:** . in column 1
- **Continuation:** + in column 1

- First line is ALWAYS title
- Last line must be .end
- Case insensitive
- Line-oriented parsing

  - Separators: spaces, =, ( ), tabs
  - Numbers: 1.23e-5, 0.0000123
  - Strings: alphanumeric sequences
  - Parameters: PARAM=VALUE format

File Structure Best Practices

Pro Tip

For advanced simulations, maintain consistent formatting and extensive commenting for complex parameter sweeps and Monte Carlo analyses.

- Use meaningful variable names for complex circuits
- Implement hierarchical commenting structure
- Version control integration
- Automated netlist generation from scripts

- Forgetting title line requirement
- Missing .end statement
- Inconsistent node numbering
- Case sensitivity issues in model names

Netlist Construction

- **Ground Node:** Must be labeled "0"
- **Node Labels:** Positive integers for compatibility
- **Every Node:** Must have a label, even two-terminal connections

Letter	Element	Syntax Example
V	Voltage Source	V1 1 0 5V
I	Current Source	I1 1 0 1mA
R	Resistor	R1 1 2 1k
C	Capacitor	C1 1 2 1p IC=0V
L	Inductor	L1 1 2 1n IC=0A
D	Diode	D1 1 2 DMODEL
Q	BJT	Q1 c b e QMODEL
M	MOSFET	M1 d g s b MMODEL
T	Transmission Line	T1 1 0 2 0 Z0=50 TD=1ns

Advanced Element Considerations

For IC Design

Pay special attention to MOSFET models (Level 1-100+), parasitic capacitances, and substrate connections

- Substrate connection critical for bulk effects
- Model level selection (BSIM3, BSIM4, PSP)
- Temperature dependencies
- Process variation parameters
- `M1 d g s sub NMOS W=10u L=0.18u` ; Explicit substrate
- `Cpd d sub 50f` ; Drain-substrate cap
- `Cgs g s 100f` ; Gate-source cap

Independent Source Specifications

- `V1 1 0 DC_VALUE AC AC_MAG AC_PHASE TRANSIENT_FUNCTION`
- `I1 1 0 DC_VALUE AC AC_MAG AC_PHASE TRANSIENT_FUNCTION`
- `VDC 1 0 5` ; 5V DC source
- `VDC 1 0 DC 5` ; Explicit DC specification
- `VAC 1 0 0 AC 1 0` ; 1V magnitude, 0° phase
- `VAC 1 0 DC 0 AC 1 90` ; 0V DC, 1V AC at 90°

Transient Analysis Functions

```
  PULSE V1 V2 TD TR TF PW
PER
```

- V1: Initial value
- V2: Peak value
- TD: Delay time
- TR: Rise time
- TF: Fall time
- PW: Pulse width
- PER: Period

```
SIN VO VA FREQ TD THETA
```

- VO: Offset voltage
- VA: Amplitude
- FREQ: Frequency
- TD: Delay
- THETA: Damping factor

- PWL (Piecewise Linear) for arbitrary waveforms
- EXP (Exponential) for RC responses
- SFFM (Single Frequency FM) for modulation analysis

Device Models

```
.MODEL MODELNAME TYPE (PARAM1=VALUE1 PARAM2=VALUE2
...)
```

- **D:** Diode models
- **NPN/PNP:** BJT models
- **NMOS/PMOS:** MOSFET models
- **NJF/PJF:** JFET models

- `.MODEL DCLAMP D(`
- `+ IS=1e-14`
- `+ RS=0.1`
- `+ CJO=1p`
- `+ VJ=0.7`
- `+ M=0.5)`

Advanced MOSFET Models for IC Design

Industry Models

BSIM3, BSIM4, PSP, HiSIM

Key Parameters: VT0, KP, LAMBDA, GAMMA, PHI, TOX, LD, WD

Process Variations: Statistical models, corner models, Monte Carlo parameters

- `.MODEL NMOS NMOS (LEVEL=49 VERSION=3.3.0`
- `+ TOX=1.4E-8 XJ=1E-7 NCH=2.3549E17 VTH0=0.3691`
- `+ K1=0.5810697 K2=4.774618E-3 K3=0.0431669`
- `+ ... [hundreds of parameters for advanced models])`

- Level 1: Simple square-law model
- Level 49: BSIM3v3 - Industry standard
- Level 54: BSIM4 - Advanced short-channel effects
- Statistical parameters for yield analysis

DC Analysis

- `.DC SRCNAME VSTART VSTOP VINCR [SRC2NAME VSTART2 VSTOP2 VINCR2]`
- `.DC V1 -5 5 0.1` ; Single source sweep
- `.DC V1 0 5 0.1 TEMP -40 125 25` ; Nested sweep with temperature
- Transfer characteristics (VTC curves)
- Bias point analysis
- Process corner verification
- Temperature coefficient analysis
- Multi-dimensional parameter sweeps
- Operating point extraction
- Small-signal parameter calculation
- Convergence analysis for complex circuits

AC Analysis

- `.AC DEC|OCT|LIN NP FSTART FSTOP`
- `.AC DEC 100 1 1G` ; Logarithmic frequency sweep
- `.AC LIN 1000 1k 100k` ; Linear frequency sweep
- **Gain-bandwidth analysis**
- **Stability analysis** (phase/gain margins)
- **Input/output impedance**
- **Frequency response optimization**

Stability Analysis

Use AC analysis with loop breaking to determine:

- Phase margin ¿ 60° for stability
- Gain margin ¿ 10dB
- Unity gain frequency

Transient Analysis

```
.TRAN TSTEP TSTOP [TSTART [TMAX]] [UIC]
.TRAN 1n 100n 0 0.1n UIC
```

- TSTEP: Print step interval
- TSTOP: Final time
- TSTART: Start time for printing (default 0)
- TMAX: Maximum internal timestep
- UIC: Use Initial Conditions

- **Rule of thumb:** TSTEP $\leq$ Period/100
- **High-speed circuits:** Consider rise/fall times
- **Oscillators:** Multiple periods for settling

- Settling time analysis
- Slew rate measurements
- Power-up sequences
- Clock jitter analysis

Initial Conditions and Advanced Transient

- `C1 1 2 1p IC=2.5V` ; Capacitor initial voltage
- `L1 1 2 1n IC=0A` ; Inductor initial current
- `.IC V(3)=1.2V V(4)=0V` ; Node initial conditions

Advanced Tip

Use .IC for complex bias networks, UIC for charge-sharing analysis

- Adaptive timestep algorithms
- Breakpoint insertion for discontinuities
- Interpolation methods for output
- Memory management for long simulations

Noise Analysis

- `.NOISE V(OUTPUT) VIN DEC 100 1 1G`
- `.NOISE V(5) VIN OCT 20 1k 1MEG 1`
- **Output node:** Where noise is measured
- **Input source:** Reference for input-referred noise
- **Applications:** LNA design, ADC noise analysis, oscillator phase noise
- Thermal noise (4kTR)
- Shot noise (2qI)
- Flicker noise (1/f)
- Induced gate noise in MOSFETs

Noise Optimization

Critical for low-noise amplifiers, precision references, and high-resolution converters

Monte Carlo and Sensitivity Analysis

- `.MC 100 TRAN 1n 100n YMAX V(out)` ; 100 runs, find max V(out)
- `.PARAM R1_VAR=GAUSS(1k,50,3)` ; Gaussian distribution
- `.SENS V(5)` ; DC sensitivity
- `.SENS V(out,5k)` ; AC sensitivity at 5kHz

Process Variation Analysis

Statistical Models: Mismatch parameters, process corners
Key Applications: Yield optimization, design centering, worst-case analysis

- Latin Hypercube Sampling
- Importance sampling for rare events
- Response surface modeling
- Design of experiments (DOE)

Output Control and Data Extraction

- `.PRINT DC V(1) V(2) I(VDD)` ; DC print
- `.PRINT AC VM(5) VP(5) VDB(5)` ; AC magnitude, phase, dB
- `.PRINT TRAN V(out) I(R1) P(M1)` ; Transient variables

- `VDB(5)` ; 20*log10(—V(5)—)
- `VP(5)` ; Phase of V(5)
- `VM(5)` ; Magnitude of V(5)
- `VR(5)` ; Real part of V(5)
- `VI(5)` ; Imaginary part of V(5)

- `@M1[gm]` ; Transconductance
- `@M1[gds]` ; Output conductance
- `@M1[id]` ; Drain current
- `@M1[vgs]` ; Gate-source voltage
- `@M1[vth]` ; Threshold voltage

Modern Graphical Output

Graphical Post-Processing

PSpice: .PROBE statement generates data for Probe viewer
Other SPICE variants: Automatic generation of plot data files
Post-processing: Python/MATLAB scripts for custom analysis

- Automated measurement functions
- Statistical data analysis
- Multi-dimensional plotting
- Export to standard formats (CSV, HDF5)

- `.MEAS TRAN trise TRIG V(out) VAL=0.1 RISE=1`
- `+ TARG V(out) VAL=0.9 RISE=1`
- `.MEAS AC bandwidth WHEN VDB(out)=-3`

SPICE Operating Modes

- `spice3 -b input.cir > output.txt` # Background batch
- `pspice input.cir` # PSpice batch
- `hspice input.sp` # HSPICE batch
- `spice3` # Start interactive session
- `spice 1 -> source input.cir` # Load circuit
- `spice 2 -> run` # Execute analyses
- `spice 3 -> setplot tran1` # Select analysis
- `spice 4 -> plot v(out)` # Generate plot
- `spice 5 -> print v(out)` # Print values
- `spice 6 -> quit` # Exit

Interactive Commands

- **run:** Execute analyses
- **stop:** Halt simulation
- **resume:** Continue simulation
- **reset:** Clear all data

- **setplot:** Choose analysis dataset
- **display:** Show available variables
- **let:** Define new variables
- **alter:** Modify circuit parameters

- Real-time parameter modification
- Interactive debugging
- Custom function definition
- Script execution within interactive mode

GUI vs Command Line Interfaces

- Visual schematic entry
- Interactive simulation control
- Real-time plotting
- Parameter sweeping tools
- Built-in calculators

- **Cadence Virtuoso**
- **Mentor Graphics ADS**
- **Synopsys HSPICE**
- **LTspice** (free)
- **Ngspice + KiCad**

Command Line Benefits

Automation: Scripting, batch processing, parameter optimization
Precision: Exact parameter control, repeatable simulations
Integration: Design flows, version control, continuous integration
Performance: Faster execution, lower memory overhead

Best Practices for IC Design

- **Initial Design:** Use GUI for rapid prototyping and visualization
- **Verification:** Command line for automated corner analysis
- **Optimization:** Scripted parameter sweeps and yield analysis
- **Documentation:** Netlist-based approach for reproducibility

- Version control systems (Git, SVN)
- Continuous integration pipelines
- Automated regression testing
- Design database management

Professional Workflow

Combine GUI for design entry with command-line automation for verification and optimization

Complete SPICE Example: Diode Clamp Circuit

Circuit Description

Application: Voltage clamping circuit with RC time constant analysis

Components: Pulse source, capacitor, diode, load resistor

```
SPICE sample circuit - diode clamp
*comment lines begin with asterisks
*independent voltage source with DC value, AC value, and
*transient square wave value:
V1 1 0 1 AC 1 pulse -10 20 0 1.e-8 1.e-8 1e-3 2e-3
*the square wave defined above has -10V to +20V extent, wit
*a period of 2 milliseconds
*capacitor for clamping
C1 1 2 1e-6
*diode for clamp - model name is dclamp
D1 2 0 dclamp
*load resistor - large enough that RC >> 2 ms
R1 2 0 1e5
*model for diode
```

Diode Clamp Example - Analysis Commands

```
*DC transfer function generated for this circuit
.DC V1 -20 20 .1
*AC frequency sweep - assumes circuit is biased with V1 = 1
.AC OCT 20 10 1e2 1e4
*frequency is swept logarithmically from 100Hz to 10000Hz
*transient analysis will show clamping
.TRAN 1e-4 8e-3 0 1e-5
*start at time zero, go for 8 ms, make internal steps 10 mi
*save print data at .1 ms intervals
*that's all folks
.end
```

- DC analysis reveals clamping threshold
- AC analysis shows frequency response limitations
- Transient analysis demonstrates dynamic clamping behavior
- Time constants validate RC ¿¿ switching period

Hierarchical Design

- `.SUBCKT OPAMP 1 2 3 4 5` ; Non-inv, Inv, Out, VDD, VSS
- `* Internal circuitry here`
- `.ENDS OPAMP`
- `X1 in1 in2 out vdd vss OPAMP` ; Instantiate subcircuit
- `.SUBCKT DIFFPAIR 1 2 3 4 PARAMS: W=10u L=0.18u`
- `M1 3 1 5 4 NMOS W={W} L={L}`
- `M2 3 2 6 4 NMOS W={W} L={L}`
- `.ENDS`
- `X1 in+ in- out vss DIFFPAIR W=20u L=0.25u`
- Multi-level subcircuit nesting
- Global parameter propagation
- Conditional circuit topology
- Library management

Process Corners and Statistical Analysis

- `.LIB ’/path/to/models’ TT` ; Typical-Typical
- `.LIB ’/path/to/models’ FF` ; Fast-Fast
- `.LIB ’/path/to/models’ SS` ; Slow-Slow
- `.LIB ’/path/to/models’ SF` ; Slow-Fast
- `.LIB ’/path/to/models’ FS` ; Fast-Slow

Statistical Analysis

Mismatch Analysis: Device-to-device variations within the same die
Process Variations: Die-to-die and wafer-to-wafer variations
Environmental Stress: Temperature, voltage, aging effects

- Systematic corner verification
- Statistical yield prediction
- Design centering optimization
- Robustness validation

Convergence Options

- `.OPTIONS GMIN=1e-12` ; Minimum conductance
- `.OPTIONS ABSTOL=1e-12` ; Absolute current tolerance
- `.OPTIONS RELTOL=1e-3` ; Relative tolerance
- `.OPTIONS VNTOL=1e-6` ; Voltage tolerance
- `.OPTIONS ITL1=100` ; DC iteration limit
- `.OPTIONS ITL2=50` ; DC transfer curve iteration limit
- `.OPTIONS ITL4=10` ; Transient iteration limit

- Poor initial guesses
- Floating nodes
- Large signal discontinuities
- Unrealistic device models
- Numerical precision limits

- Add .NODESET statements
- Use .IC for difficult bias points
- Ramp sources gradually
- Check model parameters
- Adjust solver tolerances

Advanced Solver Options

- `.OPTIONS METHOD=GEAR` ; Integration method
- `.OPTIONS MAXORD=2` ; Maximum integration order
- `.OPTIONS PIVREL=1e-3` ; Relative pivot threshold
- `.OPTIONS PIVTOL=1e-13` ; Absolute pivot threshold

- **GEAR:** Implicit multi-step method for stiff systems
- **TRAPEZOIDAL:** Second-order accurate, A-stable
- **EULER:** First-order, simple but less accurate

Convergence Strategy

Start with relaxed tolerances, then tighten for accuracy. Use source ramping and initial conditions for difficult circuits.

Simulation Strategy

1. **Hand Calculations** - Initial sizing
2. **Simplified Models** - Topology verification
3. **Detailed Models** - Performance optimization
4. **Corner Analysis** - Robustness verification
5. **Statistical Analysis** - Yield prediction

1. **Unit Cells** - Individual stages
2. **Functional Blocks** - Combined stages
3. **System Level** - Complete circuits
4. **Layout Parasitic** - Post-layout verification
5. **Silicon Correlation** - Model validation

Documentation and Version Control

Simulation Decks: Maintain versioned, commented netlist libraries

Model Management: Centralized, validated device model libraries

Results Archival: Automated result storage with simulation metadata

Performance Optimization

- **Parallel Simulation:** Multi-core utilization for corner analysis
- **Incremental Updates:** Only re-simulate changed portions
- **Checkpoint/Restart:** Resume long simulations efficiently
- **Smart Convergence:** Adaptive solver settings

- Scripted parameter sweeps
- Automated measurement extraction
- Statistical post-processing
- Design optimization loops

Modern Workflows

Integration with machine learning for design space exploration and automated optimization

Summary and Key Takeaways

Critical Knowledge for Advanced IC Design

SPICE Fundamentals: Understanding MNA, convergence, and accuracy limitations
Analysis Types: DC, AC, Transient, Noise, Monte Carlo applications
Operating Modes: Strategic use of GUI vs. command line approaches

- Advanced device modeling
- Statistical analysis interpretation
- Convergence troubleshooting
- Performance optimization

- Simulation methodology
- Documentation standards
- Version control practices
- Cross-platform compatibility

Next Steps

1. **Hands-on Practice:** Implement complex analog circuits using multiple SPICE variants
2. **Advanced Topics:** Noise optimization, mismatch analysis, yield modeling
3. **Integration:** SPICE in automated design flows and optimization loops
4. **Research Applications:** Custom models, algorithm development, emerging devices

Remember

SPICE is a tool to validate and optimize your designs, not replace fundamental analog design intuition and understanding.

Questions and Discussion