5 Circuit Simulation

5.1 Introduction

Fabricating chips is expensive and time-consuming, so designers need simulation tools to explore the design space and verify designs before they are fabricated. Simulators operate at many levels of abstraction, from process through architecture. *Process simulators* such as SUPREME predict how factors in the process recipe such as time and temperature affect device physical and electrical characteristics. *Circuit simulators* such as SPICE use device models and a circuit netlist to predict circuit voltages and currents, which indicate performance and power consumption. *Logic simulators* predict the function of digital circuits and are widely used to verify correct logical operation of designs specified in a hardware description language (HDL). *Architecture simulators* work at the level of instructions and registers to predict throughput and memory access patterns, which influence design decisions such as pipelining and cache memory organization. The various levels of abstraction offer tradeoffs between degree of detail and the size of the system that can be simulated. VLSI designers are primarily concerned with circuit and logic simulation. This chapter focuses on circuit simulation with SPICE. Section 9.3 discusses logic simulation.

Is it better to predict circuit behavior using paper-and-pencil analysis, as has been done in the previous chapters, or with simulation? VLSI circuits are complex and modern transistors have nonlinear, nonideal behavior, so simulation is necessary to accurately predict detailed circuit behavior. Even when closed-form solutions exist for delay or transfer characteristics, they are too time-consuming to apply by hand to large numbers of circuits. On the other hand, circuit simulation is notoriously prone to errors: *garbage in, garbage out* (GIGO). The simulator accepts the model of reality provided by the designer, but it is very easy to create a model that is inaccurate or incomplete. Moreover, the simulator only applies the stimulus provided by the designer and it is very easy to overlook the worst-case stimulus. In the same way that an experienced programmer doesn't expect a program to operate correctly before debugging, an experienced VLSI designer does not expect that the first run of a simulation will reflect reality. Therefore, the circuit designer needs to have a good intuitive understanding of circuit operation and should be able to predict the expected outcome before simulating. Only when expectation and simulation match can there be confidence in the results. In practice, circuit designers depend on both hand analysis and simulation, or as [Glasser85] puts it, "simulation guided through insight gained from analysis."

This chapter presents a brief SPICE tutorial by example. It then discusses models for transistors and diffusion capacitance. The remainder of the chapter is devoted to simulation techniques to characterize a process and to check performance, power, and correctness of circuits and interconnect.

5.2 A SPICE Tutorial

SPICE, a *Simulation Program with Integrated Circuit Emphasis*, was originally developed in the 1970s at Berkeley [Nagel75]. It solves the nonlinear differential equations describing components such as transistors, resistors, capacitors, and voltage sources. SPICE offers many ways to analyze circuits, but digital VLSI designers are primarily interested in *DC* and *transient* analysis that predicts the node voltages given inputs that are fixed or arbitrarily changing in time. SPICE was originally developed in FORTRAN and has some idiosyncrasies, particularly in file formats, related to its heritage. There are free versions of SPICE available on most platforms, but the commercial versions tend to offer more robust numerical convergence. In particular, HSPICE is widely used in industry because it converges well, supports the latest device and interconnect models, and has a large number of enhancements for measuring and optimizing circuits. PSPICE is another commercial version with a free limited student version. The examples throughout this section use HSPICE and generally will not run in ordinary SPICE.

While the details of using SPICE vary with version and platform, all versions of SPICE read an input file and generate a list file with results, warnings, and error messages. The input file is often called a *SPICE deck* and each line a *card* because it was once provided to a mainframe as a deck of punch cards. The input file contains a netlist consisting of components and nodes. It also contains simulation options, analysis commands, and device models. The netlist can be entered by hand or extracted from a circuit schematic or layout in a CAD program.

A good SPICE deck is like a good piece of software. It should be readable, maintainable, and reusable. Comments and white space help make the deck readable. Often the best way to write a SPICE deck is to start with a good deck that does nearly the right thing and then modify it.

The remainder of this section provides a sequence of examples illustrating the key syntax and capabilities of SPICE for digital VLSI circuits. For more detail, consult the Berkeley SPICE manual [Johnson91], the lengthy HSPICE manual, or any number of textbooks on SPICE (such as [Kielkowski95, Foty96]).

5.2.1 Sources and Passive Components

Suppose we would like to find the response of the RC circuit in Figure 5.1(a) given an input rising from 0 to 1.8 V over 50 ps. Because the RC time constant of 100 fF · 2 kΩ = 200 ps is much greater than the input rise time, we intuitively expect the output would

look like an exponential asymptotically approaching the final value of 1.8 V with a 200 ps time constant. Figure 5.2 gives a SPICE deck for this simulation and Figure 5.1(b) shows the input and output responses.

Cards beginning with * are comments. The first card of a SPICE deck must be a comment, typically indicating the title of the simulation. It is good practice to treat SPICE input files like computer programs and follow similar procedures for commenting the decks. In particular, giving the author, date, and objective of the simulation at the beginning is helpful when the deck must be revisited in the future (e.g., when a chip is in silicon debug and old simulations are being reviewed to track down potential reasons for failure).

Control cards begin with a dot (.). The `.option post` card instructs HSPICE to write the results to a file for use with a waveform viewer. The last card of a SPICE deck must be `.end`.

Each card in the netlist begins with a letter indicating the type of circuit element. Note that SPICE is case-insensitive. Common elements are given in Table 5.1. In this case the circuit consists of a voltage source named `Vin`, a resistor named `R1`, and a capacitor named `C1`. The nodes in the circuit are named `in`, `out`, and `gnd`. `gnd` is a special node name defined to be the 0 V reference. The units consist of one or two letters. The first character indicates the order of magnitude, as given in Table 5.2. The second letter indicates a unit for human convenience (such as F for farad or s for second) and is ignored by SPICE. For example, the hundred femtofarad capacitor can be expressed as `100fF`, `100f`, or simply `100e-15`.

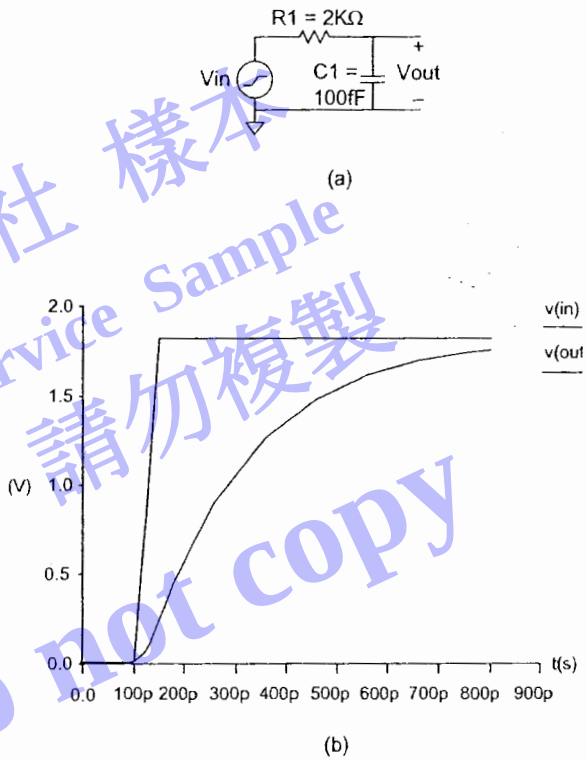

FIG 5.1 RC circuit response

The voltage source is defined as a piecewise linear (PWL) source. The waveform is specified with an arbitrary number of (time, voltage) pairs. Other common sources include DC sources and pulse sources. A DC voltage source named `Vdd` that sets node `vdd` to 2.5 V could be expressed as:

```
Vdd   vdd   gnd   2.5
```

Pulse sources are convenient for repetitive signals like clocks. The general form for a pulse source is illustrated in Figure 5.3. For example, a clock with a 1.8 V swing, 800 ps period, 100 ps rise and fall times, and 50% duty cycle (i.e., equal high and low times) would be expressed as

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

```
* rc.sp
* David_Harris@hmc.edu 2/2/03
* Find the response of RC circuit to rising input

*-------------------------------------------------------------------------
* Parameters and models
*-------------------------------------------------------------------------
.option post

*-------------------------------------------------------------------------
* Simulation netlist
*-------------------------------------------------------------------------
Vin   in    gnd   pwl   0ps 0 100ps 0 150ps 1.8 800ps 1.8
R1    in    out   2k
C1    out   gnd   100f

*-------------------------------------------------------------------------
* Stimulus
*-------------------------------------------------------------------------
.tran 20ps 800ps
.plot v(in) v(out)
.end
```

FIG 5.2 RC spice deck

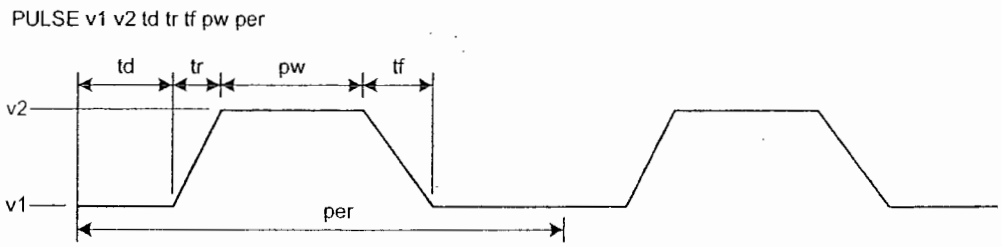

FIG 5.3 PULSE waveform

The stimulus specifies that a transient analysis (`.tran`) should be performed using a maximum step size of 20 ps for a duration of 800 ps. When plotting node voltages, the step size determines the spacing between points.

The `.plot` command generates a textual plot of the node variables specified (in this case the voltages at nodes `in` and `out`), as shown in Figure 5.4. Similarly, the `.print` statement prints the results in a multicolumn table. Both commands show the legacy of

Table 5.1 Common 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

Table 5.2 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}

FORTRAN and line printers. On modern computers with graphical user interfaces, the `.option post` command is usually preferred. It generates a file (in this case, `rc.tr0`) containing the results of the specified (transient) analysis. Then a separate graphical waveform viewer can be used to look at and manipulate the waveforms. AWAVES and COSMOSCOPE are waveform viewers compatible with HSPICE.

```
legend:
  a: v(in)
  b: v(out)

      time        v(in)
 (ab       )       0.          500.0000m         1.0000          1.5000          2.0000
                    +              +              +               +               +
   0.          0.     -2-------+-------+-------+-------+-------+-------+-------+-
  20.0000p     0.      2       +       +       +       +       +       +       +
  40.0000p     0.      2       +       +       +       +       +       +       +
  60.0000p     0.      2       +       +       +       +       +       +       +
  80.0000p     0.      2       +       +       +       +       +       +       +
 100.0000p     0.      2       +       +       +       +       +       +       +
 120.0000p  720.000m  +b       +       +      a+       +       +       +       +
 140.0000p    1.440   +   b    +       +       +       +       +   a   +       +
 160.0000p    1.800   +       +b       +       +       +       +       +      +a       +
 180.0000p    1.800   +       +    b  +       +       +       +       +      +a       +
 200.0000p    1.800  -+-------+-------+b------+-------+-------+-------+-------+a-------+-
 220.0000p    1.800   +       +       +   b   +       +       +       +      +a       +
 240.0000p    1.800   +       +       +      +b       +       +       +      +a       +
 260.0000p    1.800   +       +       +       +   b   +       +       +      +a       +
 280.0000p    1.800   +       +       +       +      b+       +       +      +a       +
 300.0000p    1.800   +       +       +       +       +b      +       +      +a       +
 320.0000p    1.800   +       +       +       +       +  b    +       +      +a       +
 340.0000p    1.800   +       +       +       +       +     b +       +      +a       +
 360.0000p    1.800   +       +       +       +       +       b       +      +a       +
 380.0000p    1.800   +       +       +       +       +       +b      +      +a       +
 400.0000p    1.800  -+-------+-------+-------+-------+-------+--b----+-------+a-------+-
 420.0000p    1.800   +       +       +       +       +       +   b   +      +a       +
 440.0000p    1.800   +       +       +       +       +       +     b +      +a       +
 460.0000p    1.800   +       +       +       +       +       +      b+      +a       +
 480.0000p    1.800   +       +       +       +       +       +       b      +a       +
 500.0000p    1.800   +       +       +       +       +       +       +b     +a       +
 520.0000p    1.800   +       +       +       +       +       +       +b     +a       +
 540.0000p    1.800   +       +       +       +       +       +       + b    +a       +
 560.0000p    1.800   +       +       +       +       +       +       +  b   +a       +
 580.0000p    1.800   +       +       +       +       +       +       +  b   +a       +
 600.0000p    1.800  -+-------+-------+-------+-------+-------+-------+---b---+a-------+-
 620.0000p    1.800   +       +       +       +       +       +       +    b  +a       +
 640.0000p    1.800   +       +       +       +       +       +       +     b +a       +
 660.0000p    1.800   +       +       +       +       +       +       +     b +a       +
 680.0000p    1.800   +       +       +       +       +       +       +     b +a       +
 700.0000p    1.800   +       +       +       +       +       +       +      b+a       +
 720.0000p    1.800   +       +       +       +       +       +       +      b+a       +
 740.0000p    1.800   +       +       +       +       +       +       +      b+a       +
 760.0000p    1.800   +       +       +       +       +       +       +      b+a       +
 780.0000p    1.800   +       +       +       +       +       +       +       ba       +
 800.0000p    1.800  -+-------+-------+-------+-------+-------+-------+-------ba-------+-
                    +              +              +               +               +
```

FIG 5.4 Textual plot of RC circuit response

5.2.2 Transistor DC Analysis

One of the first steps in becoming familiar with a new CMOS process is to look at the I-V characteristics of the transistors. Figure 5.5(a) shows test circuits for a unit (4/2 λ) nMOS transistor in a 180 nm process at V_{DD} = 1.8 V. The I-V characteristics are plotted in Figure 5.5(b) using the SPICE deck in Figure 5.6.

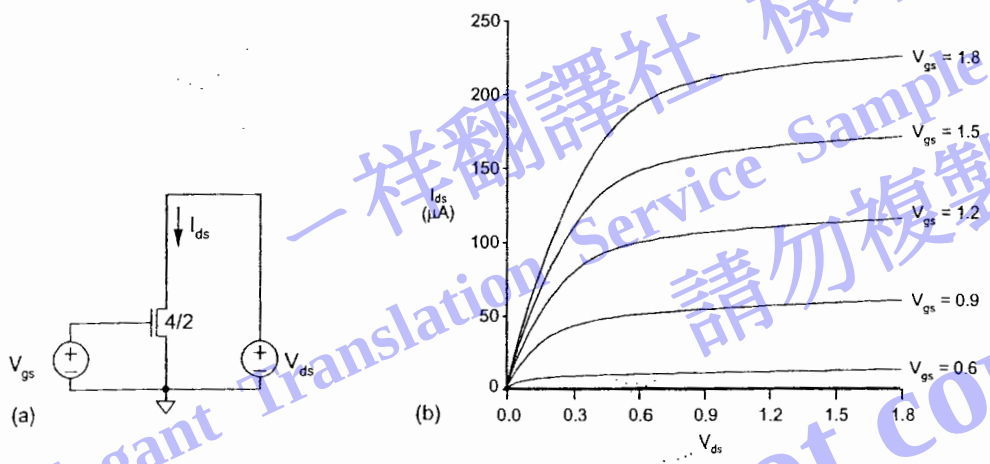

FIG 5.5 MOS I-V characteristics

```
* 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    NMOS    W=0.36u   L=0.18u

*-----------------------------------------------------------------------
* Stimulus
*-----------------------------------------------------------------------
.dc Vds 0 1.8 0.05 SWEEP Vgs 0 1.8 0.3
.end
```

FIG 5.6 MOSIV SPICE deck

`.include` reads another SPICE file from disk. In this example, it loads device models that will be discussed further in Section 5.3. The circuit uses two independent voltage sources with default values of 0 V; these voltages will be varied by the `.dc` command. The nMOS transistor is defined with the MOSFET element `M` using the syntax

```
Mname   drain   gate   source   body   type   W=<width>   L=<length>
```

The `.dc` command varies the voltage source `Vds` DC voltage from 0 to 1.8 V in increments of 0.05 V. This is repeated multiple times as `Vgs` is swept from 0 to 1.8 V in 0.3 V increments to compute many I_{ds} vs. V_{ds} curves at different values of V_{gs}.

5.2.3 Inverter Transient Analysis

Figure 5.7 shows the step response of an unloaded unit inverter, annotated with propagation delay and 20%–80% rise and fall times. Observe that significant overshoot from bootstrapping occurs because there is no load. The SPICE deck for the simulation is shown in Figure 5.8.

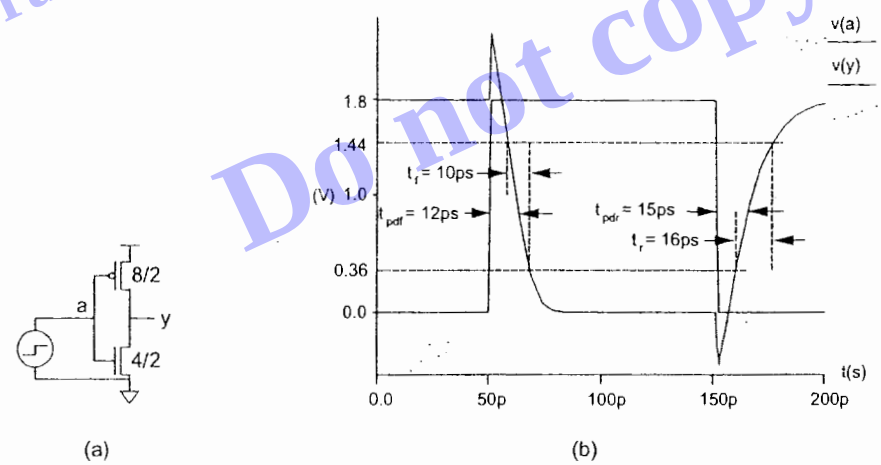

FIG 5.7 Unloaded inverter

This deck introduces the use of parameters and scaling. The `.param` statement defines a parameter named `SUPPLY` to have a value of 1.8. This is then used to set `Vdd` and the amplitude of the input pulse. If we wanted to evaluate the response at a different supply voltage, we would simply need to change the `.param` statement. The `.scale` sets a scale factor for all dimensions that would by default be measured in meters. In this case, it sets the scale to λ = 90 nm. Now the transistor widths and lengths in the inverter are specified in terms of lambda rather than in meters.

```
* 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
```

FIG 5.8 INV SPICE deck

Recall that parasitic delay is strongly dependent on diffusion capacitance, which in turn depends on the area and perimeter of the source and drain. As each diffusion region in an inverter must be contacted, the geometry resembles that of Figure 2.9(a). The diffusion width equals the transistor width and the diffusion length is 5 λ. Thus, the area of the source and drain are AS = AD = $5W\lambda^2$ and the perimeters are PS = PD = $(2W + 10)\lambda$. Note that the + sign in the first column of a card indicates that it is a continuation of the previous card. These dimensions are also affected by the scale factor.

5.2.4 Subcircuits and Measurement

One of the simplest measures of a process's inherent speed is the fanout-of-4 inverter delay. Figure 5.9(a) shows a circuit to measure this delay. The nMOS and pMOS transistor sizes (in multiples of a unit 4/2 λ transistor) are listed below and above each gate, respectively. *X3* is the inverter under test and *X4* is its load, which is four times larger than *X3*. To first order, these two inverters would be sufficient. However, the delay of *X*3 also depends on the input slope, as discussed in Section 4.2.5.1. One way to obtain a realistic input slope is to drive node c with a pair of FO4 inverters *X1* and *X2*. Also, as discussed in Section 4.2.5.4, the input capacitance of *X4* depends not just on its C_{gs} but also

on C_{gd}. C_{gd} is Miller-multiplied as node e switches and would be effectively doubled if e switched instantaneously. When e is loaded with *X5*, it switches at a slower, more realistic rate, slightly reducing the effective capacitance presented at node d by *X4*. The waveforms in Figure 5.9(b) are annotated with the rising and falling delays.

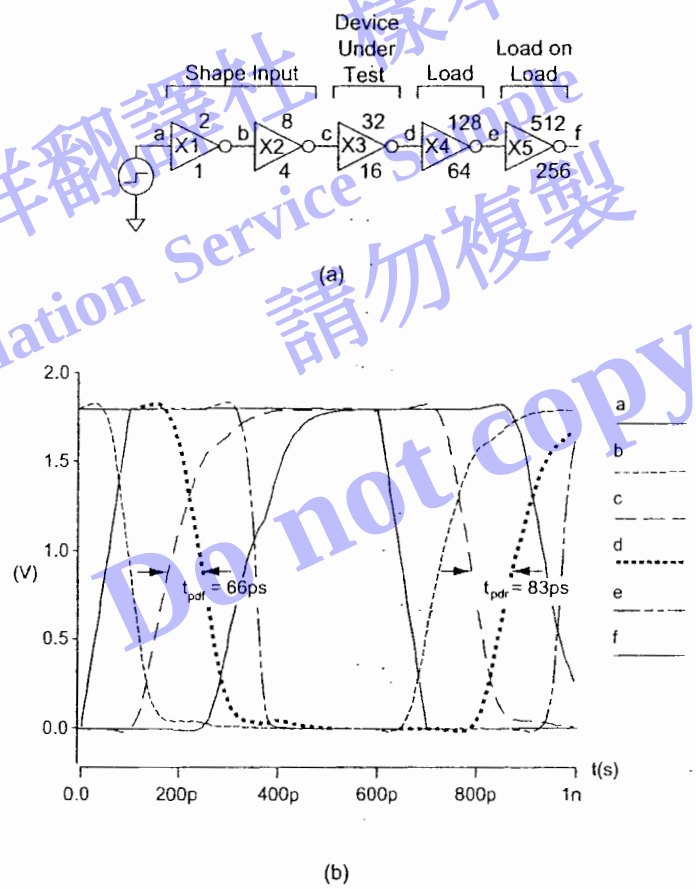

FIG 5.9 Fanout-of-4 inverters

SPICE decks are easier to read and maintain when common circuit elements are captured as subcircuits. For example, the deck in Figure 5.10 computes the FO4 inverter delay using an inverter subcircuit.

The `.global` statement defines vdd and gnd as global nodes that can be referenced from within subcircuits. The inverter is declared as a subcircuit with two terminals: a and y. It also accepts two parameters specifying the width of the nMOS and pMOS transis-

```
* fo4.sp
*-----------------------------------------------------------------------
* Parameters and models
*-----------------------------------------------------------------------
.param SUPPLY=1.8
.param H=4
.option scale=90n
.include '../models/tsmc180/models.sp'
.temp 70
.option post
*-----------------------------------------------------------------------
* Subcircuits
*-----------------------------------------------------------------------
.global vdd gnd
.subckt inv a y N=4 P=8
M1    y     a     gnd    gnd     NMOS    W='N'   L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2    y     a     vdd    vdd     PMOS    W='P'   L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends
*-----------------------------------------------------------------------
* Simulation netlist
*-----------------------------------------------------------------------
Vdd   vdd   gnd   'SUPPLY'
Vin   a     gnd   PULSE   0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
X1    a     b     inv                 * shape input waveform
X2    b     c     inv     M='H'       * reshape input waveform
X3    c     d     inv     M='H**2'    * device under test
X4    d     e     inv     M='H**3'    * load
X5    e     f     inv     M='H**4'    * load on load
*-----------------------------------------------------------------------
* Stimulus
*-----------------------------------------------------------------------
.tran 1ps 1000ps
.measure tpdr                                 * rising prop delay
+     TRIG v(c)    VAL='SUPPLY/2' FALL=1
+     TARG v(d)    VAL='SUPPLY/2' RISE=1
.measure tpdf                                 * falling prop delay
+     TRIG v(c)    VAL='SUPPLY/2' RISE=1
+     TARG v(d)    VAL='SUPPLY/2' FALL=1
.measure tpd param='(tpdr+tpdf)/2'            * average prop delay
.measure trise                                * rise time
+     TRIG v(d)    VAL='0.2*SUPPLY' RISE=1
+     TARG v(d)    VAL='0.8*SUPPLY' RISE=1
.measure tfall                                * fall time
+     TRIG v(d)    VAL='0.8*SUPPLY' FALL=1
+     TARG v(d)    VAL='0.2*SUPPLY' FALL=1
.end
```

FIG 5.10 FO4 SPICE deck

tors; these parameters have default values of 4 and 8, respectively. The source and drain area and perimeter are functions of the transistor widths. HSPICE evaluates functions given inside single quotation marks. The functions can include parameters, constants, parentheses, +, -, *, /, and ** (raised to a power).

The simulation netlist contains the power supply, input waveform, and four inverters. Each inverter is subcircuit (X) element. As N and P are not specified, each uses the default size. The M parameter multiplies all the currents in the subcircuit by the factor given, equivalent to M elements wired in parallel. In this case, the fanouts are expressed in terms of a parameter H. Thus, *X2* has the capacitance and output current of 4 unit inverters, while *X3* is equivalent to 16. Another way to model the inverters would have been to use the N and P parameters:

```
X1   a   b   inv   N=4      P=8      * shape input waveform
X2   b   c   inv   N=16     P=32     * reshape input waveform
X3   c   d   inv   N=64     P=128    * device under test
X4   d   e   inv   N=256    P=512    * load
X5   e   f   inv   N=1024   P=2048   * load on load
```

However, a transistor of four times unit width does not have exactly the same input capacitance or output current as four unit inverters tied in parallel, so the M parameter is usually preferred.

In this example, the subcircuit declaration and simulation netlist are part of the SPICE deck. When working with a standard cell library, it is common to keep subcircuit declarations in their own files and reference them with a `.include` statement instead. When the simulation netlist is extracted from a schematic or layout CAD system, it is common to put the netlist in a separate file and `.include` it as well.

The `.measure` statement measures simulation results and prints them in the listing file. The deck measures the rising propagation delay t_{pdr} as the difference between the time that the input c first falls through $V_{DD}/2$ and the time that the output d first rises through $V_{DD}/2$. TRIG and TARG indicate the trigger and target events between which delay is measured. The `.measure` statement can also be used to compute functions of other measurements. For example, the average FO4 inverter propagation delay t_{pd} is the mean of t_{pdr} and t_{pdf}, i.e., t_{pd} = 75 ps. The 20%–80% rise time is t_r = 94 ps and the fall time is t_f = 67 ps.

5.2.5 Optimization

In many examples, we have assumed that a *P*/*N* ratio of 2:1 gives approximately equal rise and fall delays. The FO4 inverter simulation showed that a ratio of 2:1 gives rising delays that are slower than the falling delays because the pMOS mobility is less than half that of the nMOS. You could repeatedly run simulations with different default values of P to find the ratio for equal delay. HSPICE has built-in optimization capabilities that will automatically tweak parameters to achieve some goal and report what parameter value gave the best results. Figure 5.11 shows a modified version of the FO4 inverter simulation using the optimizer.

```
* fo4opt.sp
*-----------------------------------------------------------------------
* Parameters and models
*-----------------------------------------------------------------------
.param SUPPLY=1.8
.option scale=90n
.include '../models/tsmc180/models.sp'
.temp 70
.option post
*-----------------------------------------------------------------------
* Subcircuits
*-----------------------------------------------------------------------
.global vdd gnd
.subckt inv a y N=4 P=8
M1   y   a   gnd   gnd   NMOS   W='N'   L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2   y   a   vdd   vdd   PMOS   W='P'   L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends
*-----------------------------------------------------------------------
* Simulation netlist
*-----------------------------------------------------------------------
Vdd   vdd   gnd   'SUPPLY'
Vin   a     gnd   PULSE    0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
X1    a     b     inv      P='P1'                * shape input waveform
X2    b     c     inv      P='P1'    M=4         * reshape input waveform
X3    c     d     inv      P='P1'    M=16        * device under test
X4    d     e     inv      P='P1'    M=64        * load
X5    e     f     inv      P='P1'    M=256       * load on load
*-----------------------------------------------------------------------
* Optimization setup
*-----------------------------------------------------------------------
.param P1=optrange(8,4,16)          * search from 4 to 16, guess 8
.model optmod opt itropt=30         * maximum of 30 iterations
.measure bestratio param='P1/4'     * compute best P/N ratio
*-----------------------------------------------------------------------
* Stimulus
*-----------------------------------------------------------------------
.tran 1ps 1000ps SWEEP OPTIMIZE=optrange RESULTS=diff MODEL=optmod
.measure tpdr                                * rising propagation delay
+     TRIG v(c)   VAL='SUPPLY/2' FALL=1
+     TARG v(d)   VAL='SUPPLY/2' RISE=1
.measure tpdf                                * falling propagation delay
+     TRIG v(c)   VAL='SUPPLY/2' RISE=1
+     TARG v(d)   VAL='SUPPLY/2' FALL=1
.measure tpd param='(tpdr+tpdf)/2' goal=0   * average prop delay
.measure diff param='tpdr-tpdf' goal = 0    * diff between delays
.end
```

FIG 5.11 FO4OPT SPICE deck

The subcircuits *X1–X4* override their default pMOS widths to use a width of `P1` instead. In the optimization setup, the difference of t_{pdr} and t_{pdf} is measured. The goal of the optimization will be to drive this difference to 0. To do this, `P1` may be varied from 4 to 16, with an initial guess of 8. The optimizer may use up to 30 iterations to find the best value of `P1`. Because the nMOS width is fixed at 4, the best *P/N* ratio is computed as `P1`/4. The transient analysis includes a `SWEEP` statement containing the parameter to vary, the desired result, and the number of iterations.

HSPICE determines that the *P/N* ratio for equal rise and fall delay is 3.6:1, giving a rising and falling delay of 84 ps. This is slower than the 2:1 ratio provides and requires large pMOS transistors that consume area and power, so such a high ratio is seldom used.

A similar scenario is to find the *P/N* ratio that gives lowest average delay. By changing the `.tran` card to use `RESULTS=tpd`, we find a best ratio of 1.4:1 with rising, falling, and average propagation delays of 87, 59, and 73 ps, respectively. Whenever you do an optimization, it is important to consider not only the optimum but also the sensitivity to deviations from this point. Further simulation finds that *P/N* ratios of anywhere from 1.2:1 to 1.7:1 all give an average propagation delay of 73 ps, there is no need to slavishly stick to the 1.4:1 "optimum." The best *P/N* ratio in practice is a compromise between using smaller pMOS devices to save area and power and using larger devices to achieve more nearly equal rise/fall times and avoid the hot electron reliability problems induced by very slow rising edges in circuits with weak pMOS transistors. *P/N* ratios are discussed further in Section 6.2.1.6.

5.2.6 Other HSPICE Commands

The full HSPICE manual fills over 2000 pages and includes many more capabilities than can be described here. A few of the most useful additional commands are covered in this section. Section 5.3 describes transistor models and library calls, while Section 5.6 discusses modeling interconnect with lossy transmission lines.

`.option accurate`

Tighten integration tolerances to obtain more accurate results. Useful for oscillators and high-gain analog circuits or when results seem fishy.

`.option autostop`

Conclude simulation when all `.measure` results are obtained rather than continuing for the full duration of the `.tran` card. This can substantially reduce simulation time.

`.temp 0 70 125`

Repeat the simulation three times at temperatures of 0°, 70°, and 125° C. Device models may contain information about how changing temperature changes device performance.

`.op`

Print the voltages, currents, and transistor bias conditions at the DC operating point.

5.3 Device Models

Most of the examples in Section 5.2 included a file containing transistor models. SPICE provides a wide variety of MOS transistor models with various tradeoffs between complexity and accuracy. Level 1 and Level 3 models were historically important, but they are no longer adequate to accurately model very small modern transistors. BSIM models are more accurate and are presently the most widely used. Some companies use their own proprietary models. This section briefly describes the main features of each of these models. It also describes how to model diffusion capacitance and how to run simulations in various process corners. The model descriptions are intended only as an overview of the capabilities and limitations of the models; refer to a SPICE manual for a much more detailed description if one is necessary.

5.3.1 Level 1 Models

The SPICE Level 1, or Shichman-Hodges Model [Shichman68] is closely related to the Shockley model described in EQ (2.10), enhanced with channel length modulation and the body effect. The basic current model is:

$$I_{ds} = \begin{cases} 0 & V_{gs} < V_t \quad \text{cutoff} \\ \mathrm{KP}\dfrac{W_{\mathrm{eff}}}{L_{\mathrm{eff}}}\left(1+\mathrm{LAMBDA}\cdot V_{ds}\right)\left(V_{gs}-V_t-\dfrac{V_{ds}}{2}\right)V_{ds} & V_{ds} < V_{gs}-V_t \quad \text{linear} \\ \dfrac{\mathrm{KP}}{2}\dfrac{W_{\mathrm{eff}}}{L_{\mathrm{eff}}}\left(1+\mathrm{LAMBDA}\cdot V_{ds}\right)\left(V_{gs}-V_t\right)^2 & V_{ds} > V_{gs}-V_t \quad \text{saturation} \end{cases} \tag{5.1}$$

The parameters from the SPICE model are given in ALL CAPS. Notice that β is written instead as $\mathrm{KP}(W_{\mathrm{eff}}/L_{\mathrm{eff}})$, where KP is a model parameter playing the role of k' from EQ (2.7). W_{eff} and L_{eff} are the effective width and length, as described in Section 2.4.8. The LAMBDA term models channel length modulation (see Section 2.4.2).

The threshold voltage is modulated by the source-to-body voltage V_{sb} through the body effect (see Section 2.4.3). For nonnegative V_{sb}, the threshold voltage is

$$V_t = \mathrm{VTO} + \mathrm{GAMMA}\left(\sqrt{\mathrm{PHI}+V_{sb}} - \sqrt{\mathrm{PHI}}\right) \tag{5.2}$$

Notice that this is identical to EQ (2.30), where VTO is the "zero-bias" threshold voltage V_{t0}, GAMMA is the body effect coefficient γ, and PHI is the surface potential ϕ_s.

The gate capacitance is calculated from the oxide thickness TOX. The default gate capacitance model in HSPICE is adequate for finding the transient response of digital circuits. More elaborate models exist that capture nonreciprocal effects that are important for analog design.

Level 1 models are useful for teaching because they are easy to correlate with hand analysis, but are too simplistic for modern design. Figure 5.12 gives an example of a Level 1 model card illustrating the card syntax. The card also includes terms to compute the diffusion capacitance, as described in Section 5.3.4.

```
.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)
```

FIG 5.12 Sample Level 1 MODEL card

5.3.2 Level 2 and 3 Models

The SPICE Level 2 and 3 models add effects of velocity saturation, mobility degradation, subthreshold conduction, and drain-induced barrier lowering. The Level 2 model is based on the Grove-Frohman equations [Frohman69], while the Level 3 model is based on empirical equations that provide similar accuracy and faster simulation times and better convergence. However, these models still do not provide good fits to the measured I-V characteristics of modern transistors.

5.3.3 BSIM Models

The Berkeley Short-Channel IGFET[1] Model (BSIM) is a very elaborate model that is now widely used in circuit simulation. The models are derived from the underlying device physics but use an enormous number of parameters to fit the behavior of modern transistors. BSIM versions 1, 2, 3v3, and 4 are implemented as SPICE levels 13, 39, 49, and 54, respectively.

BSIM version 3v3 requires an entire book [Cheng99] to describe the model. It includes over 100 parameters and the device equations span 27 pages. It is quite good for digital circuit simulation except that it does not model gate leakage. Features of the model include:

- Continuous and differentiable I-V characteristics across subthreshold, linear, and saturation regions for good convergence
- Sensitivity of parameters such as V_t to transistor length and width
- Detailed threshold voltage model including body effect and drain-induced barrier lowering
- Velocity saturation, mobility degradation, and other short-channel effects
- Multiple gate capacitance models
- Diffusion capacitance and resistance models

[1]IGFET in turn stands for Insulated-Gate Field Effect Transistor, a synonym for MOSFET.

BSIM version 4 was fairly new at the time this book was being written and adds support for gate leakage and other effects of very thin gates. As these effects are rapidly becoming more important and better characterized, the designer should check with the process engineers before blindly trusting gate leakage models.

Some device parameters such as threshold voltage change significantly with device dimensions. BSIM models can be *binned* with different models covering different ranges of length and width specified by LMIN, LMAX, WMIN, and WMAX parameters. For example, one model might cover transistors with channel lengths from 0.18–0.25 μm, another from 0.25–0.5 μm, and a third from 0.5–5 μm. SPICE will complain if a transistor does not fit in one of the bins.

As the BSIM models are so complicated, it is impractical to derive closed-form equations for propagation delay, switching threshold, noise margins, etc., from the underlying equations. However, it is not difficult to find these properties through circuit simulation. Section 5.4 will show simple simulations to plot the device characteristics over the regions of operation that are interesting to most digital designers and to extract effective capacitance and resistance averaged across the switching transition. The simple RC model continues to give the designer important insight about the characteristics of logic gates.

5.3.4 Diffusion Capacitance Models

The p–n junction between the source or drain diffusion and the body forms a reverse-biased diode. We have seen that the diffusion capacitance determines the parasitic delay of a gate and depends on the area and perimeter of the diffusion. HSPICE provides a number of methods to specify this geometry, controlled by the ACM (Area Calculation Method) parameter, which is part of the transistor model card. The model card must also have values for junction and sidewall diffusion capacitance, as described in Section 2.2.2.3. The diffusion capacitance model is common across most device models including Levels 1–3 and BSIM.

By default, HSPICE models use ACM=0. In this method, the designer must specify the area and perimeter of the source and drain of each transistor. For example, the dimensions of each diffusion region from Figure 2.9 is listed in Table 5.3 (in units of λ^2 for area or λ for perimeter). A SPICE description of the shared contacted diffusion case is shown in Figure 5.13, assuming `.option scale` is set to the value of λ.

Table 5.3 Diffusion area and perimeter

	AS1 / AD2	PS1 / PD2	AD1 / AS2	PD1 / PS2
(a) Isolated contacted diffusion	$W \cdot 5$	$2 \cdot W + 10$	$W \cdot 5$	$2 \cdot W + 10$
(b) Shared contacted diffusion	$W \cdot 5$	$2 \cdot W + 10$	$W \cdot 3$	$W + 6$
(c) Merged uncontacted diffusion	$W \cdot 5$	$2 \cdot W + 10$	$W \cdot 1.5$	$W + 3$

```
* (b): Shared contacted diffusion
M1    mid    b    bot    gnd    NMOS    W='w'    L=2
+ AS='w*5' PS='2*w+10' AD='w*3' PD='w+6'
M2    top    a    mid    gnd    NMOS    W='w'    L=2
+ AS='w*3' PS='w+6' AD='w*5' PD='2*w+10'
```

FIG 5.13 SPICE model of transistors with shared contacted diffusion

The SPICE models also should contain parameters CJ, CJSW, PB, PHP, MJ, and MJSW. Assuming the diffusion is reverse-biased and the area and perimeter are specified, the diffusion capacitance between source and body is computed as described in Section 2.3.3.

$$C_{sb} = \text{AS} \cdot \text{CJ} \cdot \left(1 + \frac{V_{sb}}{\text{PB}}\right)^{-MJ} + \text{PS} \cdot \text{CJSW} \cdot \left(1 + \frac{V_{sb}}{\text{PHP}}\right)^{-MJSW} \tag{5.3}$$

The drain equations are analogous, with S replaced by D in the model parameters.

The BSIM3 models offer a similar area calculation model (ACM = 10) that takes into account the different sidewall capacitance on the edge adjacent to the gate. Note that the PHP parameter is renamed to PBSW to be more consistent.

$$C_{sb} = \text{AS} \cdot \text{CJ} \cdot \left(1 + \frac{V_{sb}}{\text{PB}}\right)^{-MJ} + (\text{PS} - W) \cdot \text{CJSW} \cdot \left(1 + \frac{V_{sb}}{\text{PBSW}}\right)^{-MJSW} + W \cdot \text{CJSWG} \cdot \left(1 + \frac{V_{sb}}{\text{PBSWG}}\right)^{-MJSWG} \tag{5.4}$$

If the area and perimeter are not specified, they default to 0 in ACM = 0 or 10, grossly underestimating the parasitic delay of the gate. HSPICE also supports ACM = 1, 2, 3, and 12 that provide nonzero default values when the area and perimeter are not specified. Check your models and read the HSPICE documentation carefully.

The diffusion area and perimeter is also used to compute the junction leakage current. However, this current is generally negligible compared to subthreshold leakage in modern devices.

5.3.5 Design Corners

Engineers often simulate circuits in multiple design corners to verify that operation across variations in device characteristics and environment. HSPICE includes the .lib card that makes changing libraries easy. For example, the deck in Figure 5.14 runs three simulations on the step response of an unloaded inverter in the TT, FF, and SS corners.

```
* corner.sp
* Step response of unloaded inverter across process corners

*-------------------------------------------------------------------------
* Parameters and models
*-------------------------------------------------------------------------
.option scale=90n
.param SUP=1.8 * Must set before calling .lib
.lib '../models/tsmc180/opconditions.lib' TT
.option post

*-------------------------------------------------------------------------
* Simulation netlist
*-------------------------------------------------------------------------
Vdd   vdd   gnd   'SUPPLY'
Vin   a     gnd   PULSE    0 'SUPPLY' 200ps 0ps 0ps 500ps 1000ps
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 1000ps
.alter
.lib '../models/tsmc180/opconditions.lib' FF
.alter
.lib '../models/tsmc180/opconditions.lib' SS
.end
```

FIG 5.14 CORNER SPICE deck

The deck first sets SUP to the nominal supply voltage of 1.8 V. It then invokes the `.lib` card that reads in the library specifying the TT conditions. In the stimulus, the `.alter` statement is used to repeat the simulation with changes. In this case, the design corner is changed. Altogether, three simulations are performed and three sets of waveforms are generated for the three design corners.

The library file is given in Figure 5.15. Depending on what library was specified, the temperature is set (in degrees Celsius, with `.temp`) and the V_{DD} value SUPPLY is calculated from the nominal SUP. The library loads the appropriate nMOS and pMOS transistor models. A fast process file might have lower nominal threshold voltages V_{t0}, greater lateral diffusion L_D, and lower diffusion capacitance values.

```
* opconditions.lib
* For TSMC 180 nm process

* TT: Typical nMOS, pMOS, voltage, temperature
.lib TT
.temp 70
.param SUPPLY='SUP'
.include 'modelsTT.sp'
.endl TT

* SS: Slow nMOS, pMOS, low voltage, high temperature
.lib SS
.temp 125
.param SUPPLY='0.9 * SUP'
.include 'modelsSS.sp'
.endl SS

* FF: Fast nMOS, pMOS, high voltage, low temperature
.lib FF
.temp 0
.param SUPPLY='1.1 * SUP'
.include 'modelsFF.sp'
.endl FF

* FS: Fast nMOS, Slow pMOS, typical voltage and temperature
.lib FS
.temp 70
.param SUPPLY='SUP'
.include 'modelsFS.sp'
.endl FS

* SF: Slow nMOS, Fast pMOS, typical voltage and temperature
.lib SF
.temp 70
.param SUPPLY='SUP'
.include 'modelsSF.sp'
.endl SF
```

FIG 5.15 OPCONDITIONS library

5.4 Device Characterization

Modern SPICE models have so many parameters that the designer cannot easily read key performance characteristics from the model files. A more convenient approach is to run a set of simulations to extract the effective resistance and capacitance, the fanout-of-4 inverter delay, the I-V characteristics, and other interesting data. This section describes these simulations and compares the results across a variety of CMOS processes.