

The Look-Up Table Approach and Implementation in a Circuit Simulator

Mahesh Patil

Microelectronics Group

Department of Electrical Engineering

Indian Institute of Technology, Bombay

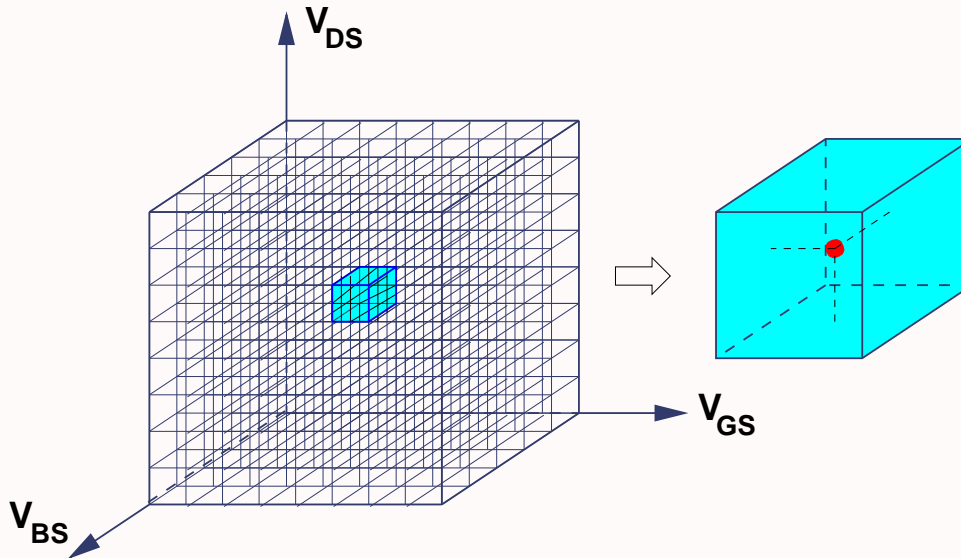
`mbpatil@ee.iitb.ac.in`

Ack.: D. Vinay Kumar, N. R. Mohapatra
Prof. V. R. Rao, Prof. J. Vasi
DST, Govt. of India
Intel, USA

Outline

- LUT approach
- Terminal charge extraction
- Exploring novel technologies with LUT approach
- Evaluation of a device model (NQS effects)
- Implementation in a circuit simulator

The LUT Approach

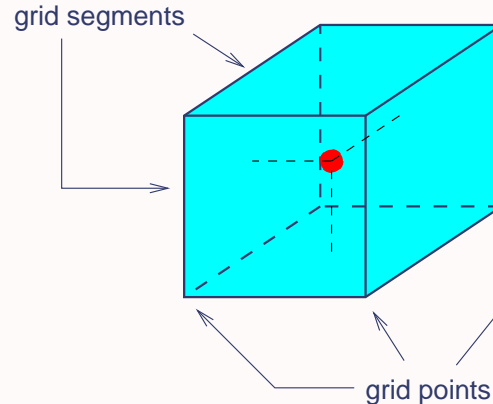


- Want to find $f(x, y, z)$, given the values of f at all grid points, $f(x_i, y_j, z_k)$.
- Since this only involves interpolation, substantial amount of time is saved in the function evaluation phase.

The LUT Approach

- The function f could be $I_d, I_g, I_b, Q_d, Q_g, Q_b$.
- If the device capacitances are to be approximated by some average values, Q_d, Q_g, Q_b need not be computed.
- However, in modern VLSI circuits, the device capacitances must be treated accurately, thus requiring the computation of the terminal charges, Q_d, Q_g, Q_b as a function of bias voltages.

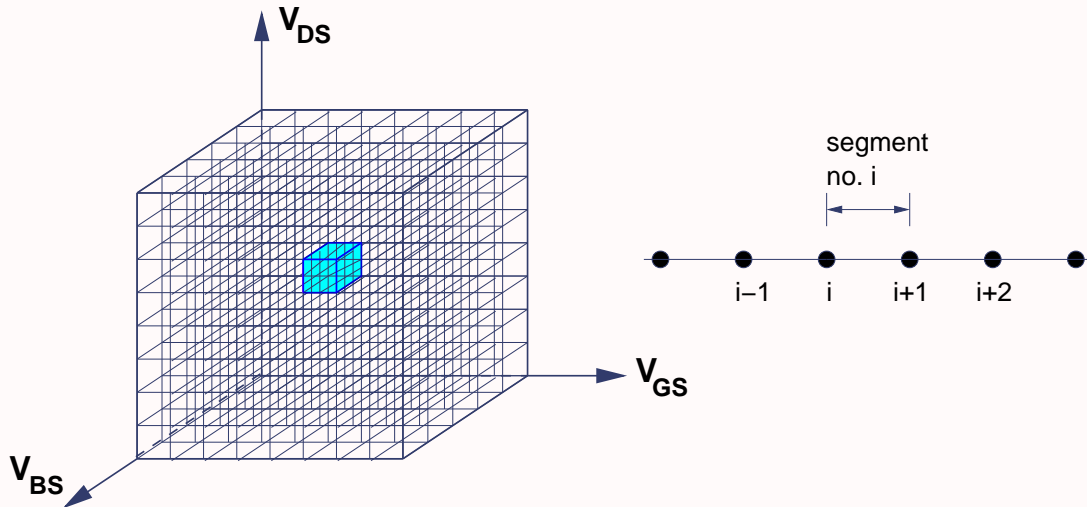
The LUT Approach



1. Construct suitable interpolating functions along each grid segment (e.g., $a_0 + a_1x + a_2x^2 + a_3x^3$). Store the coefficients.
2. Find the indices i, j, k for the given point (i.e., locate the “little cube” which contains the given point).
3. Use “Table I” or “Table II” formula in [1] to compute $f, \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$.

[1] P. B. L. Meijer, *IEEE T. Circuits and Systems*, vol. 37, p. 335, March 1990.

Step 1: coefficients in the local 1-d functions



e.g., $\phi^{(i)} = a_0^{(i)} + a_1^{(i)} x + a_2^{(i)} x^2$.

To compute $a_0^{(i)}$, $a_1^{(i)}$, $a_2^{(i)}$, table points (a) i , $i+1$, $i+2$ or (b) $i-1$, i , $i+1$ may be used.

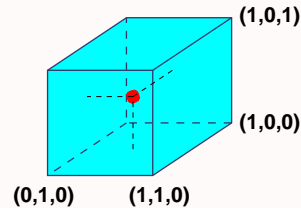
There are $N_y N_z (N_x - 1)$ segments along the x direction, and each has three coefficients.

Also, need to consider the other two directions and six different functions I_d , I_g , I_b , Q_d , Q_g , Q_b .

However, this is a one-time computation.

Step 2: Locate the “little” cube: i.e., find i_1, i_2, i_3 .

Step 3: Interpolation [Meijer, 1990]



(note: $(0, 0, 0) \equiv (i_1, i_2, i_3)$, $(1, 0, 0) \equiv (i_1 + 1, i_2, i_3)$, etc.)

Define $\bar{\tau} = \begin{pmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix}$ where $\tau_1 = 0, 1$, $\tau_2 = 0, 1$, $\tau_3 = 0, 1$.

The table points involved are $F(i_1 + \tau_1, i_2 + \tau_2, i_3 + \tau_3)$.

The function is to be interpolated at (x_1, x_2, x_3) , using the eight F values.

“Table I” formula:

$$F(x_1, x_2, x_3) = \sum_{\tau=0}^1 \left[(-2) F_{i_1+\tau_1, i_2+\tau_2, i_3+\tau_3} + \phi_{i_1, i_2+\tau_2, i_3+\tau_3}^{(1)} + \phi_{i_1+\tau_1, i_2, i_3+\tau_3}^{(2)} + \phi_{i_1+\tau_1, i_2+\tau_2, i_3}^{(3)} \right] \\ \times \prod_{k=1}^3 \left(\tau_k - \frac{x_{k, i_k+1} - x_k}{x_{k, i_k+1} - x_{k, i_k}} \right)$$

Extraction of terminal charges

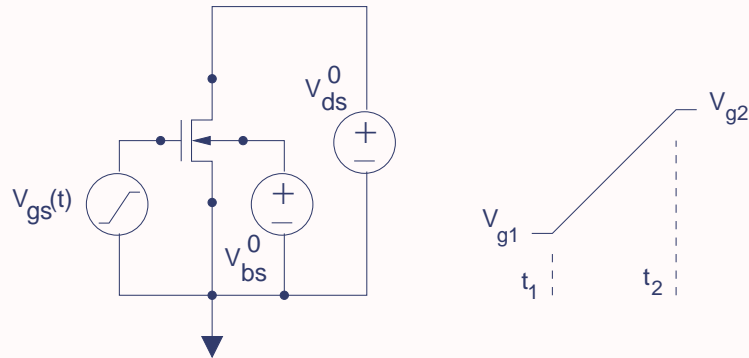
Quasi-Static approximation:

$$\begin{aligned} I_x(t) &= I_X(V_{GS}(t), V_{DS}(t), V_{BS}(t)) + \frac{\partial Q_X}{\partial t} \\ &= I_X(V_{GS}(t), V_{DS}(t), V_{BS}(t)) + \frac{\partial Q_X}{\partial V_{GS}} \frac{dV_{GS}}{dt} + \frac{\partial Q_X}{\partial V_{DS}} \frac{dV_{DS}}{dt} + \frac{\partial Q_X}{\partial V_{BS}} \frac{dV_{BS}}{dt} \end{aligned}$$

- I_X and Q_X are “DC-like” quantities.
- I_X = the DC current that would result if the voltages were frozen at $V_{GS}(t)$, $V_{DS}(t)$, $V_{BS}(t)$.
- Q_X = the terminal charge as a function of $(V_{GS}(t), V_{DS}(t), V_{BS}(t))$ *irrespective* of the past history.
- If I_X and Q_X are derived from measurements or device simulation and used as look-up tables, an *exact* QS model is obtained. (We will assume that no error is made in interpolation.)

Extraction of terminal charges:

(a) transient analysis



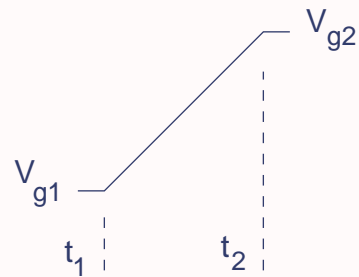
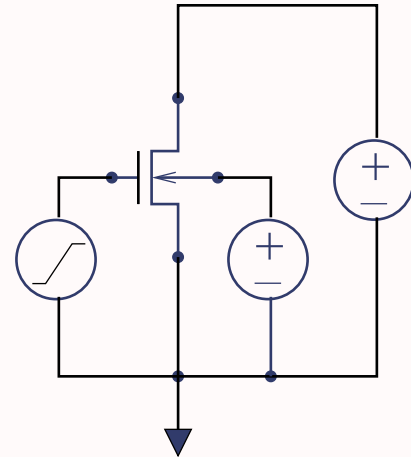
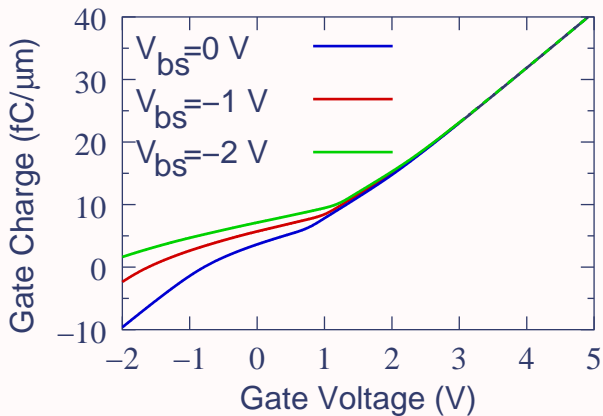
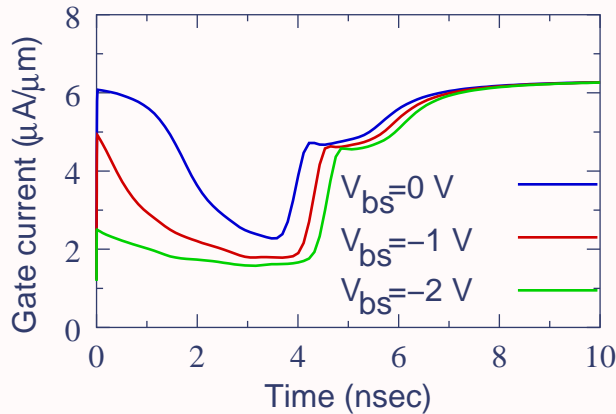
Example: Hold V_{BS} and V_{DS} constant, and vary V_{GS} linearly.

$$I_x(t) = I_X(V_{GS}(t), V_{DS}(t), V_{BS}(t)) + \frac{\partial Q_X}{\partial V_{GS}} \frac{dV_{GS}}{dt} + \frac{\partial Q_X}{\partial V_{DS}} \frac{dV_{DS}}{dt} + \frac{\partial Q_X}{\partial V_{BS}} \frac{dV_{BS}}{dt}$$

$$\Rightarrow I_D(t) - I_D(V_{GS}(t), V_{DS}^0, V_{BS}^0) = \frac{\partial Q_D}{\partial V_{GS}} \times \left(\frac{V_{g2} - V_{g1}}{t_2 - t_1} \right) \Rightarrow \frac{\partial Q_D}{\partial V_{GS}}$$

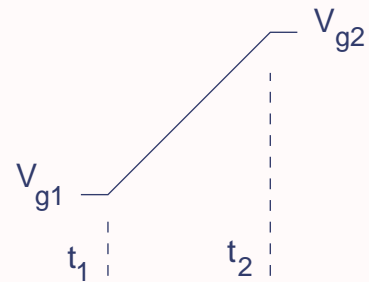
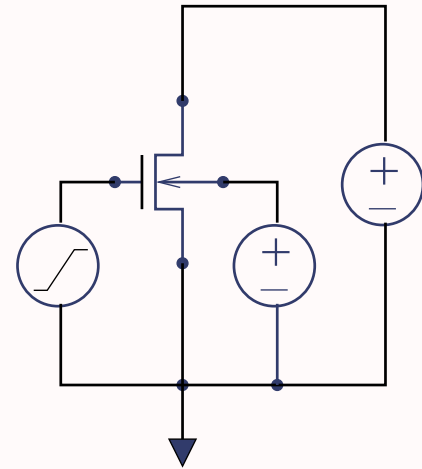
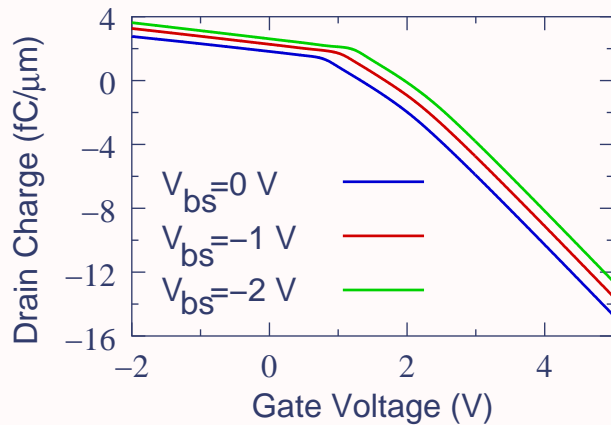
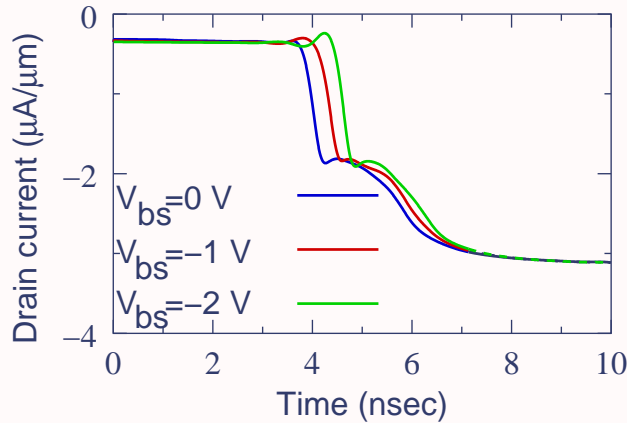
- Now integrate $\frac{\partial Q_D}{\partial V_{GS}}$ w. r. t. V_{GS} to obtain $Q_D(V_{GS}, V_{DS}^0, V_{BS}^0)$.
- Repeat for different sets of (V_{DS}^0, V_{BS}^0) .
- Assign the integration constants so that the charge is independent of the path.

Computation of Q_{gate}



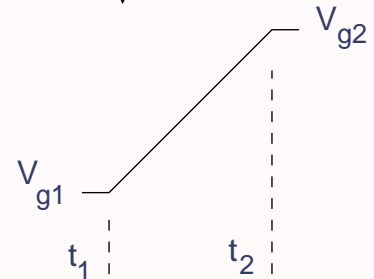
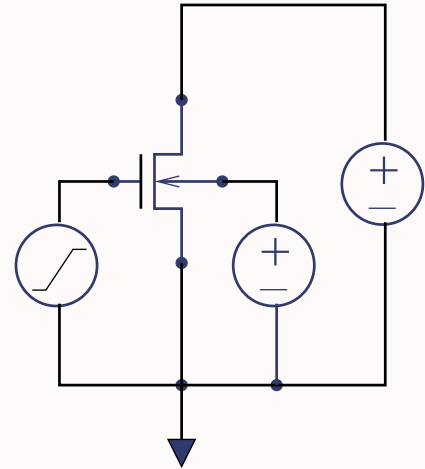
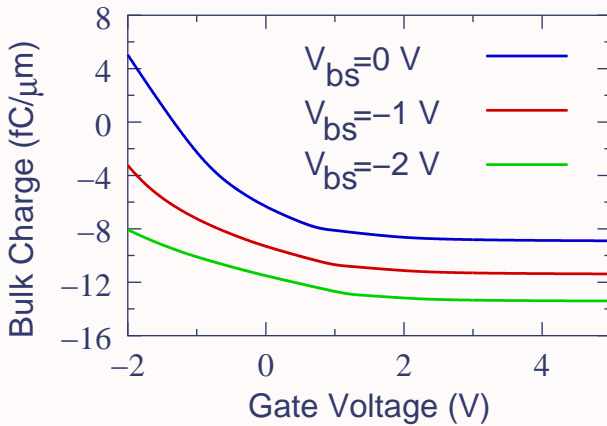
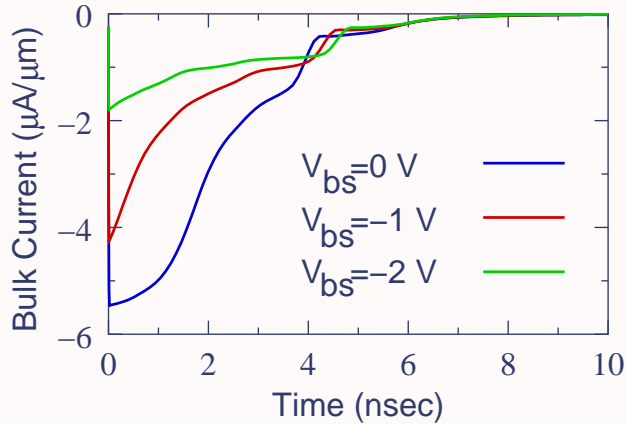
$(L=2 \mu m, V_{DS}=1 V)$

Computation of Q_{drain}



$(L=2\ \mu\text{m}, V_{\text{DS}}=1\text{ V})$

Computation of Q_{bulk}



$(L=2\ \mu\text{m}, V_{\text{DS}}=1\text{ V})$

Extraction of terminal charges:

(b) small-signal parameters

Example: Measure Y_{ig} ($i=g, d, b, s$) at a given bias condition, $(V_{GS}^0, V_{DS}^0, V_{BS}^0)$. Use

$$\mathbf{Y}_{dg}(V_{GS}^0, V_{DS}^0, V_{BS}^0) = g_{dg}(V_{GS}^0, V_{DS}^0, V_{BS}^0) + j\omega C_{dg}(V_{GS}^0, V_{DS}^0, V_{BS}^0),$$

$$C_{dg}(V_{GS}^0, V_{DS}^0, V_{BS}^0) = \frac{\partial Q_D}{\partial V_{GS}}(V_{GS}^0, V_{DS}^0, V_{BS}^0).$$

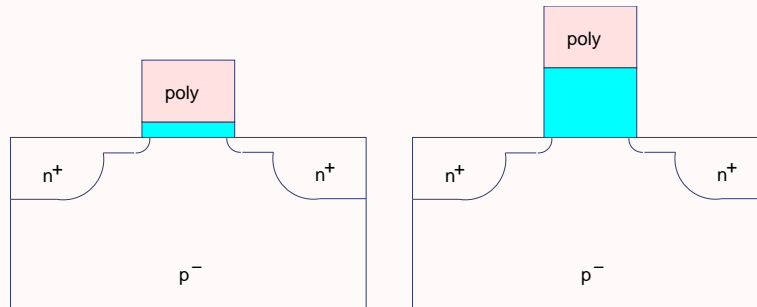
- Repeat for different values of V_{GS}^0 .
- Now integrate $\frac{\partial Q_D}{\partial V_{GS}}$ w. r. t. V_{GS} to obtain $Q_D(V_{GS}^0, V_{DS}^0, V_{BS}^0)$.

(Ref.: E. P. Vandamme *et al*, *Solid-State Electron.*, vol. 46, no. 3, p. 353, March 2002)

Exploring Novel Technologies:

- Accurate device models are generally not available for novel (“first-time”) devices.
- Model development can take a very long time.
- The LUT approach offers a quick way to estimate circuit performance without going through the model development step.
- Limitations:
 - (a) NQS effects are ignored, but this is usually not a concern in digital circuits.
 - (b) Statistical variations cannot be analyzed.
 - (c) Need to create separate tables for different channel lengths (and also for different channel widths if narrow-width effects are significant).

High- K gate MOS transistors



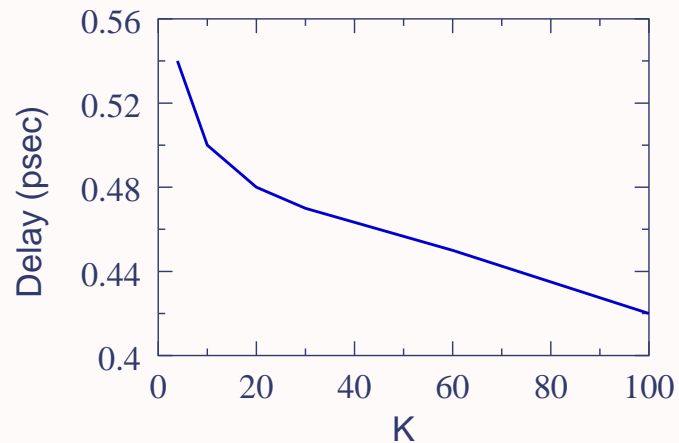
- With device scaling, the insulator thickness needs to be scaled down \Rightarrow increase in tunneling currents and power dissipation.
- Since $C_i = \frac{K\epsilon_0}{t_i}$, the insulator thickness can be increased, by increasing K in the same proportion \Rightarrow high- K gate dielectrics
- However, as $K \uparrow$, $V_T \downarrow$, fringing \uparrow , DIBL \uparrow , subthreshold slope \uparrow , $\frac{I_{on}}{I_{off}} \downarrow$.
- These effects are far too complex, making it difficult to estimate the impact of increasing K on circuit performance.
- A good candidate for LUT!

[1] N. R. Mohapatra *et al*, *IEEE Trans. Electron Devices*, vol. 49, May 2002, p. 826.

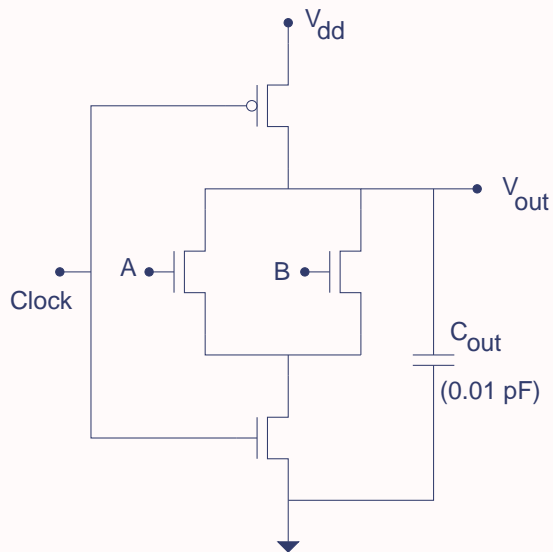
LUT simulation of High- K MOS transistor circuits

- Look-up tables are generated using MEDICI for various K values, keeping the equivalent oxide thickness constant (1.5 nm).
- The gate length is $L=70$ nm in all cases; all other details (doping densities, junction depths) are also kept identical.
- Circuits are simulated with different K values, and comparisons are made.

High- K circuits: Ring Oscillator (N=25)

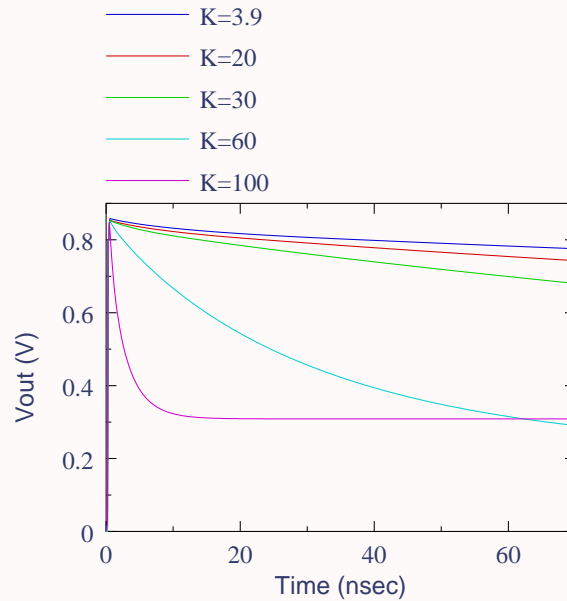
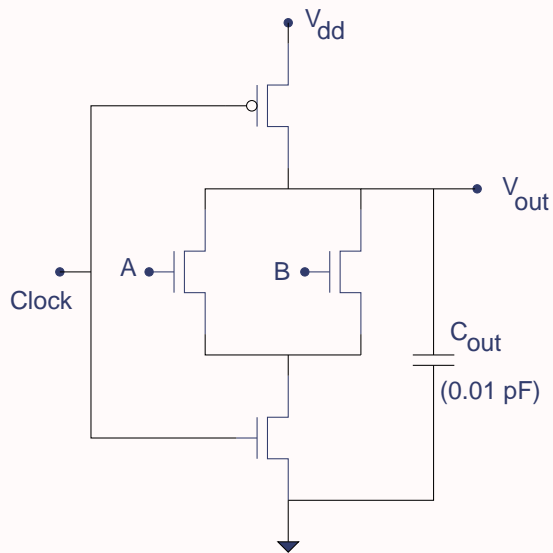


High- K Circuits: Dynamic NOR Gate

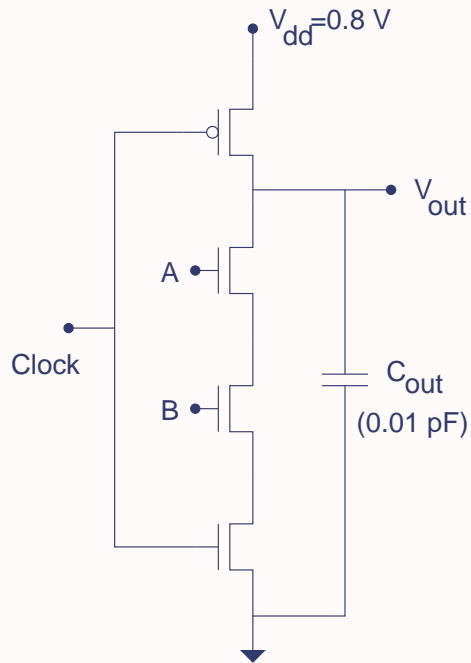


Initially, Clock is made low, and C_{out} is charged to V_{dd} . Subsequently, Clock is made high. C_{out} discharges if A or B is high; if not, it should hold the charge, i.e., V_{out} should remain equal to V_{dd} .

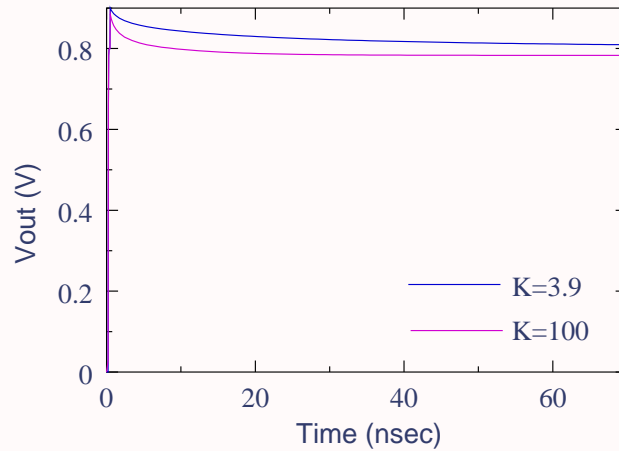
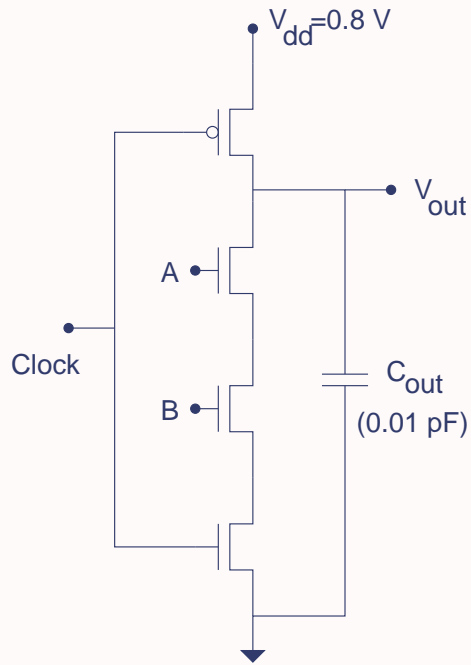
High- K Circuits: Dynamic NOR Gate



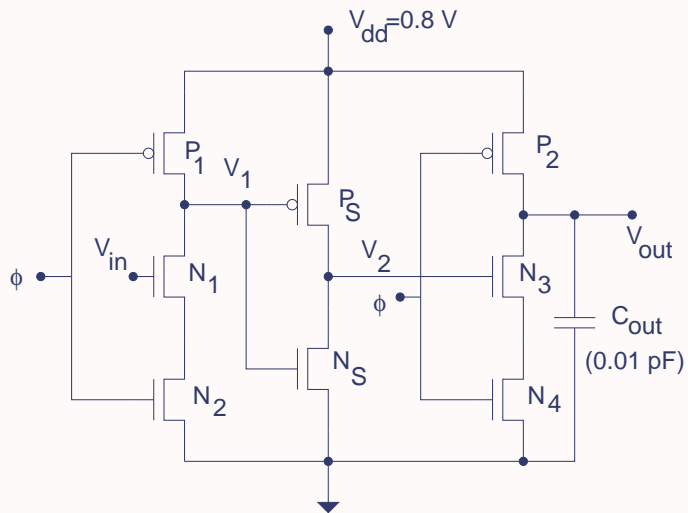
High- K Circuits: Dynamic NAND Gate



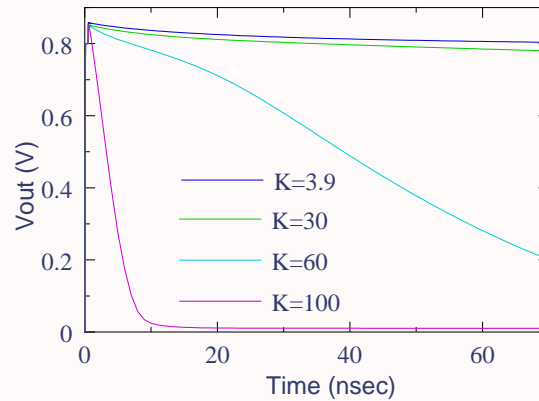
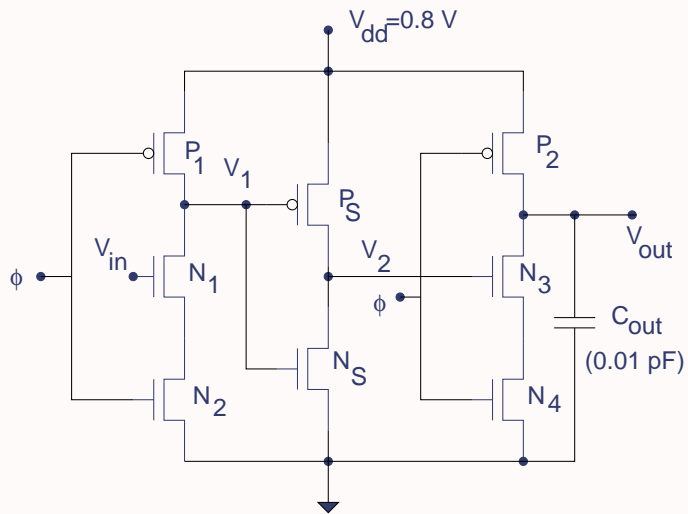
High- K Circuits: Dynamic NAND Gate



High- K Circuits: Domino Gate



High- K Circuits: Domino Gate



Evaluation of NQS effects

- LUT is an exact QS model; thus, LUT results can be used as a guide for developing an accurate QS model.
- Actual experimental or device simulation results have NQS effects embedded within; thus, comparison of these with the LUT results will help in developing an accurate NQS model.
- Existing analytical QS models can be tested against the LUT results.
- Existing analytical NQS models can be tested against the experimental or device simulation results.

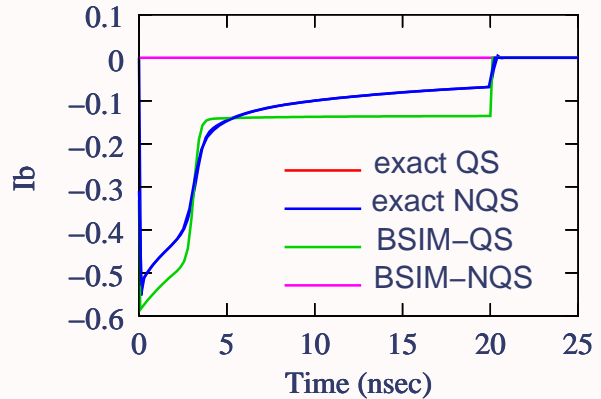
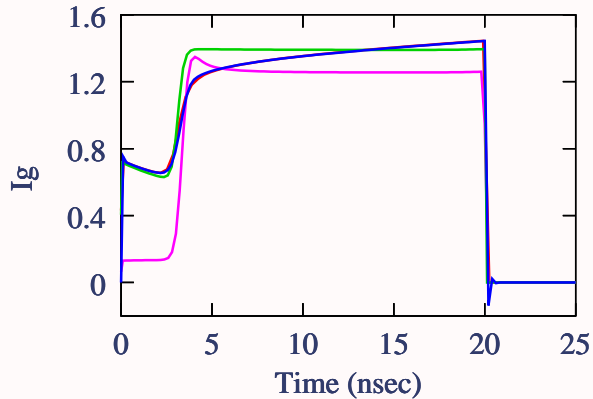
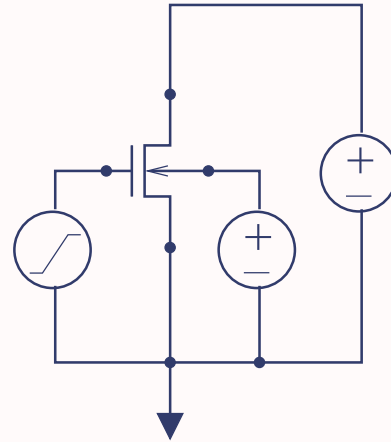
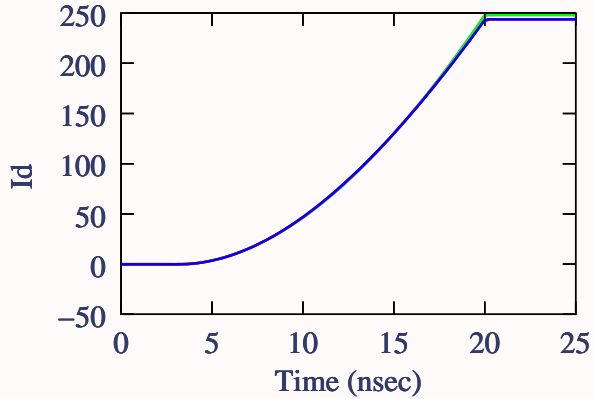
Comparison with BSIM3v3

- For a device with $L=2 \mu\text{m}$, look-up tables are generated using ISE \Rightarrow LUT model
- Using ISE-EXTRACT, BSIM3v3 parameters were extracted for the same device (20 I - V and 3 C - V curves are internally generated by ISE DESSIS for this purpose).
- The following comparisons are made
 - (a) “Slow” transient (NQS effects are negligible)
 - (b) “Fast” transient (NQS effects are significant)
 - (c) y parameters
- Note: LUT \equiv “exact” QS model, ISE \equiv “exact” NQS model.

Evaluation of NQS effects: “Slow” transients

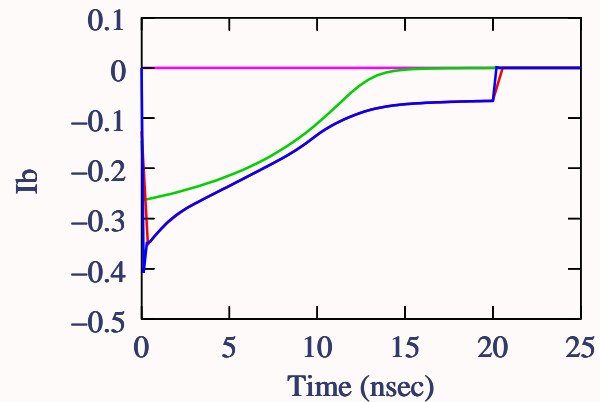
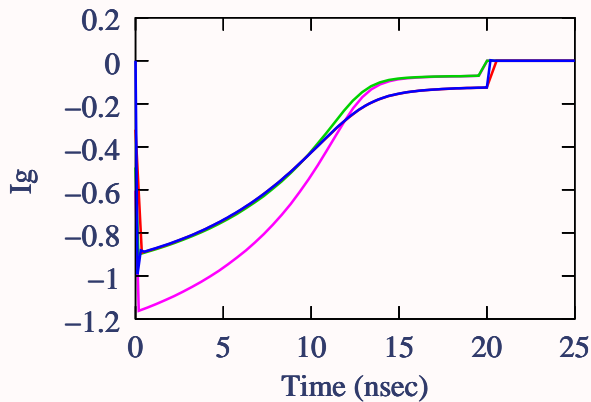
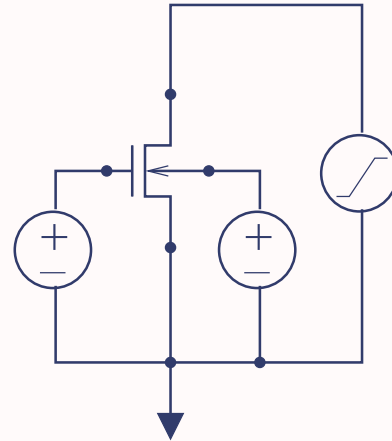
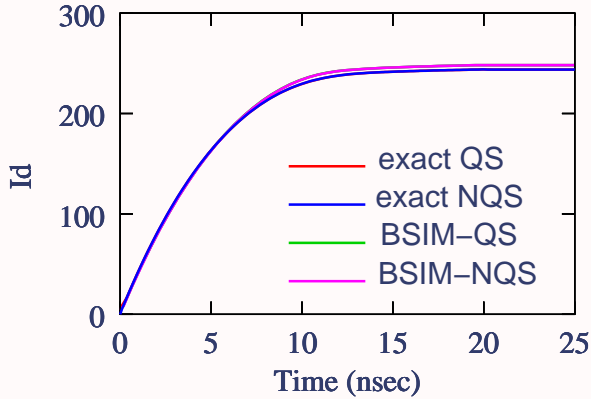
- (1) We would expect the exact QS and exact NQS results to coincide.
- (2) We would also expect the BSIM3v3-QS and BSIM3v3-NQS results to coincide
 - (a) with each other and
 - (b) with the exact results.

“Slow” V_G transient



($V_D^0=4\text{ V}$, $V_B^0=0\text{ V}$, V_G rises from 0 V to 4 V in 20 nsec. All currents are in $\mu\text{A}/\mu\text{m}$.)

“Slow” V_D transient

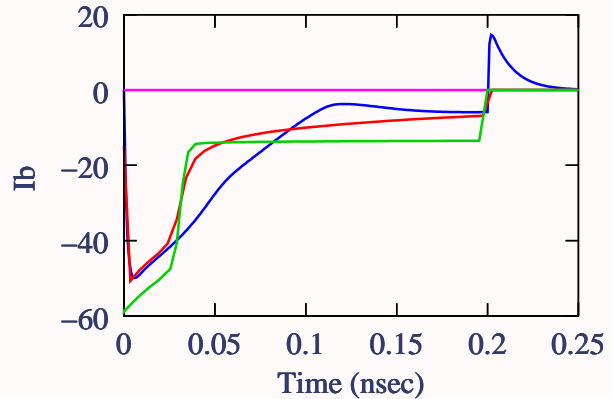
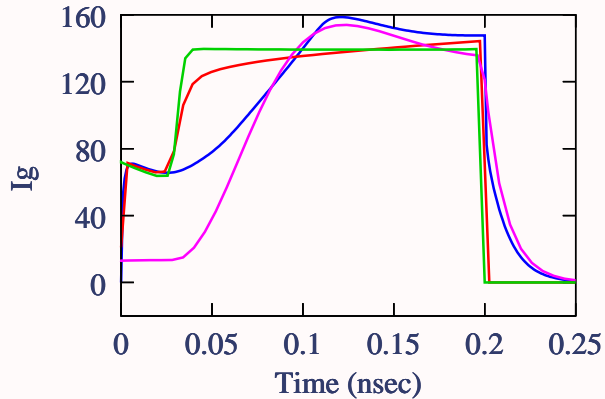
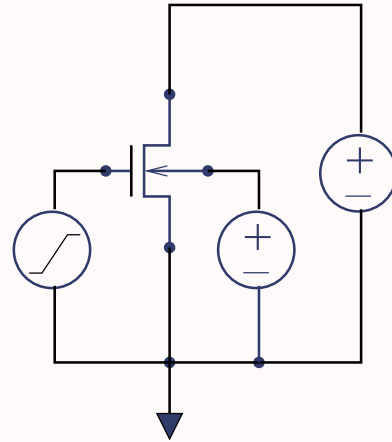
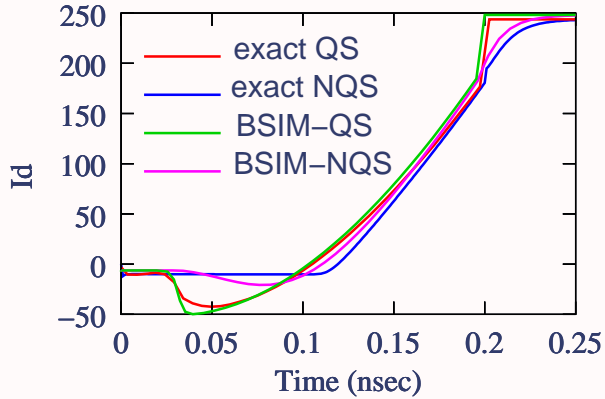


($V_G^0=4$ V, $V_B^0=0$ V, V_D rises from 0 V to 4 V in 20 nsec. All currents are in $\mu A/\mu m$.)

Evaluation of NQS effects: “Fast” transients

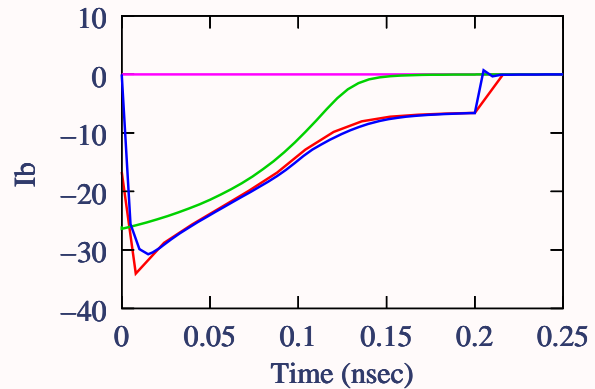
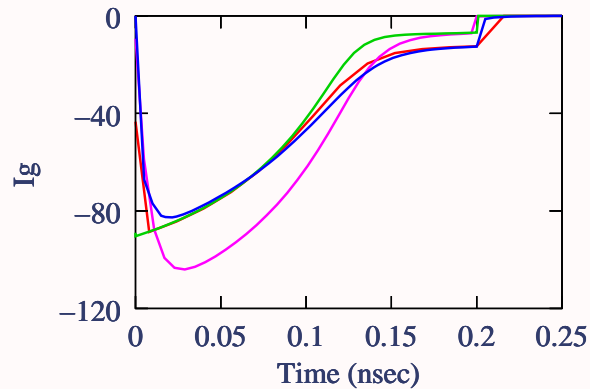
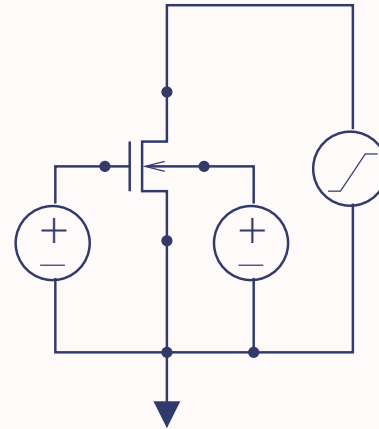
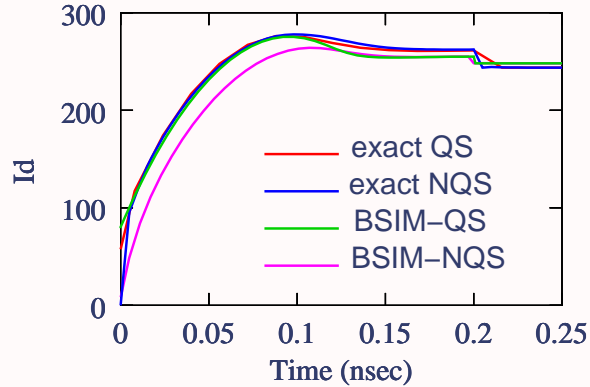
- (1) We would expect some difference between the exact QS and exact NQS results.
- (2) However, we would expect the BSIM3v3-QS results to coincide with the exact QS results,
and
- (3) We would expect the BSIM3v3-NQS results to coincide with the exact NQS results.

“Fast” V_G transient



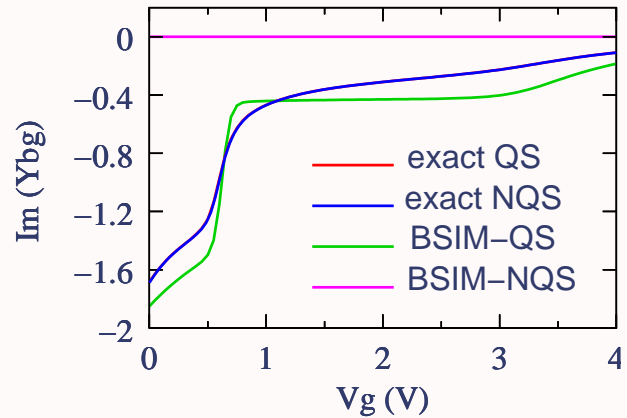
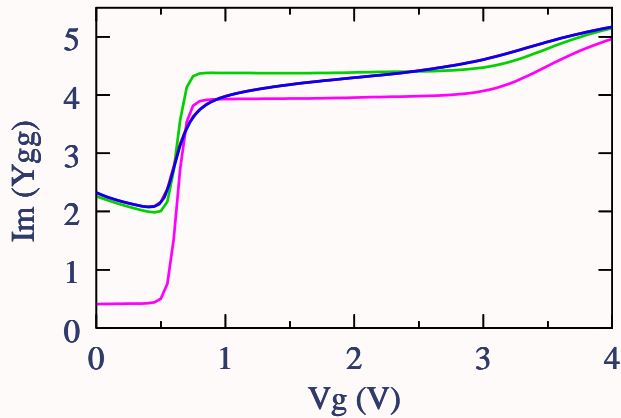
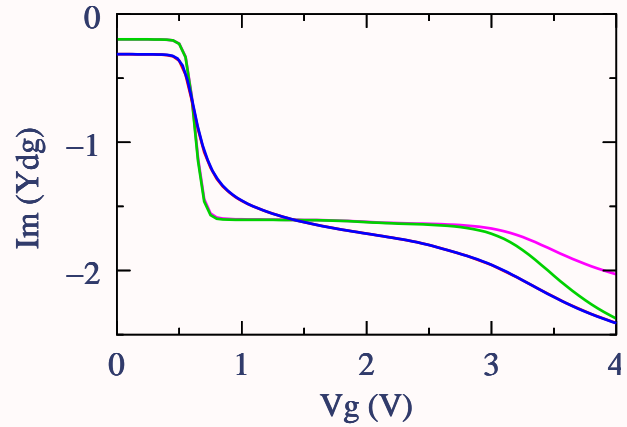
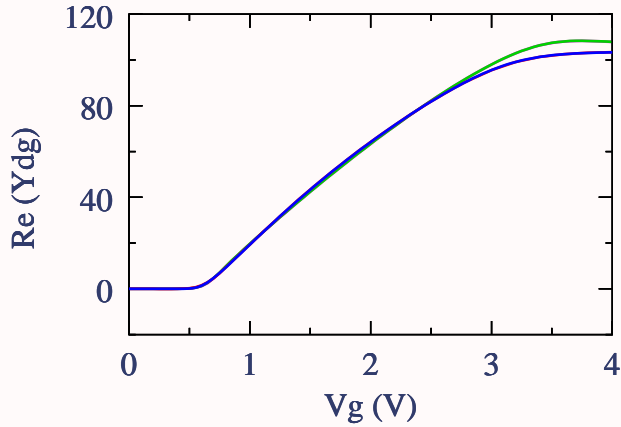
($V_D^0=4$ V, $V_B^0=0$ V, V_G rises from 0 V to 4 V in 0.2 nsec. All currents are in $\mu A/\mu m$.)

“Fast” V_D transient



($V_G^0=4$ V, $V_B^0=0$ V, V_D rises from 0 V to 4 V in 0.2 nsec. All currents are in $\mu A/\mu m$.)

Small-signal parameters



($V_{DS}=2$ V, $V_{BS}=0$ V, $f=100$ MHz. All parameters are in $\mu\text{A}/\mu\text{m}$.)

Implementation of the LUT approach in a circuit simulator:

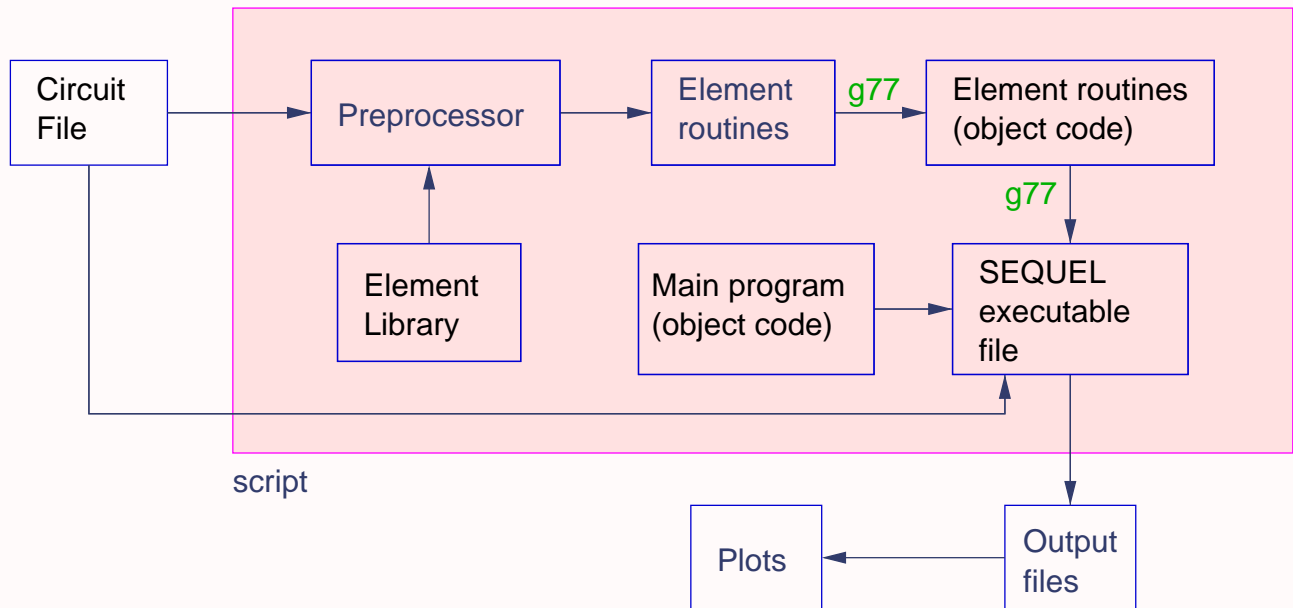
- Compute the “segment coefficients” for each segment and for each of the variables involved: “preprocessing”
- In each Newton-Raphson iteration,
 - (a) Evaluate the function values and their derivatives (w.r.t. V_{GS} , V_{DS} , V_{BS}) using interpolation.
 - (b) Use these to assign the appropriate elements in the “system matrix” (for the circuit being simulated).

SEQUEL (a Solver for circuit Equations with User-defined Elements)

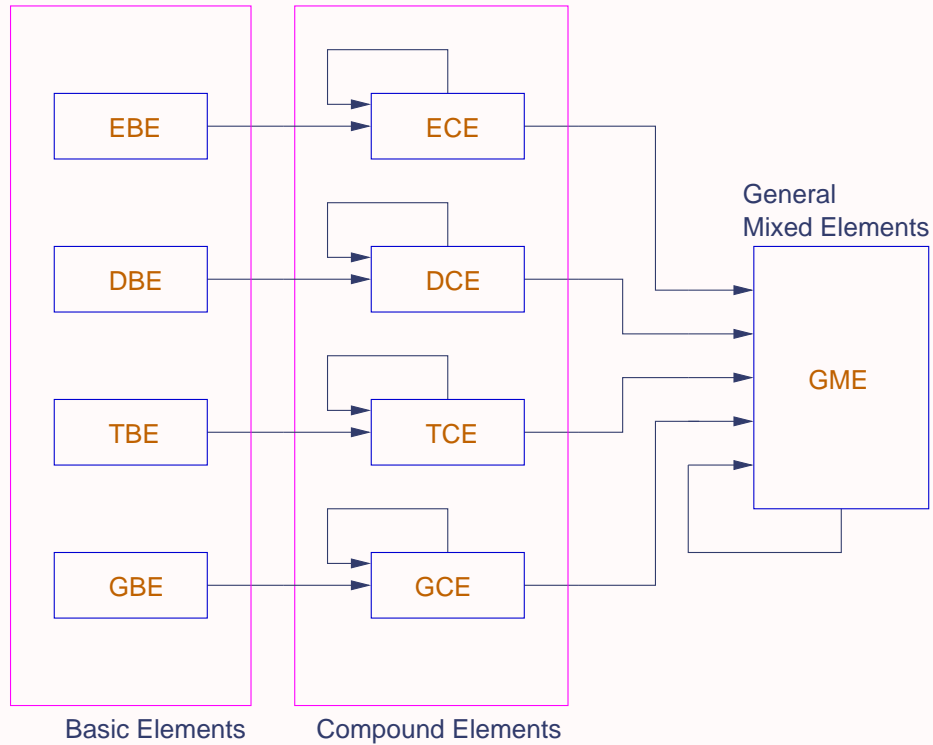
- Allows user-defined elements
- DC, transient, small-signal, noise analysis
- Sensitivity analysis
- Efficient “steady-state waveform” computation
- Mixed-signal simulation
- Switched capacitor circuits
- Power electronic circuits
- Electrothermal simulation
- Perfectly “general” elements are also possible (mechanical, thermal etc.)
- Runs on GNU/Linux (6.2 and up) and Solaris
- Free !!

<http://www.ee.iitb.ac.in/~microel/faculty/mbp/sequel1.html>

SEQUEL: Block Diagram



SEQUEL: Library Structure



(E: Electrical, D: Digital, T: Thermal, G: General)

SEQUEL: writing a template

The general form of a basic element equation is:

$$f(x_1, x_2, \dots, x_N) = \frac{dx_j}{dt} \quad \text{or}$$

$$f(x_1, x_2, \dots, x_N) = 0$$

The element template must convey to the program,

- (i) What variables do the functions depend on?
- (ii) How to evaluate the function for a given set of x_1, x_2 , etc.?
- (iii) How to evaluate the Jacobian entries $\frac{\partial f_i}{\partial x_k}$?

SEQUEL: resistor template

```
ebe name=r
nodes: p n
rparms: r=0.0
be_v: v1=p-n
no_functions=1
f_1: v1[1] current[fixed]
fortran:
source:
    if (l_dc.or.l_trns.or.l_startup) then
        r=rparm(1)
        v1=v(1)
        if (l_function) f(1)=v1-r*cur(1)
        if (l_jacobian) then
            dfdi(1,1)=-r
        end if
    end if
endfortran
endebe
```

SEQUEL: capacitor template

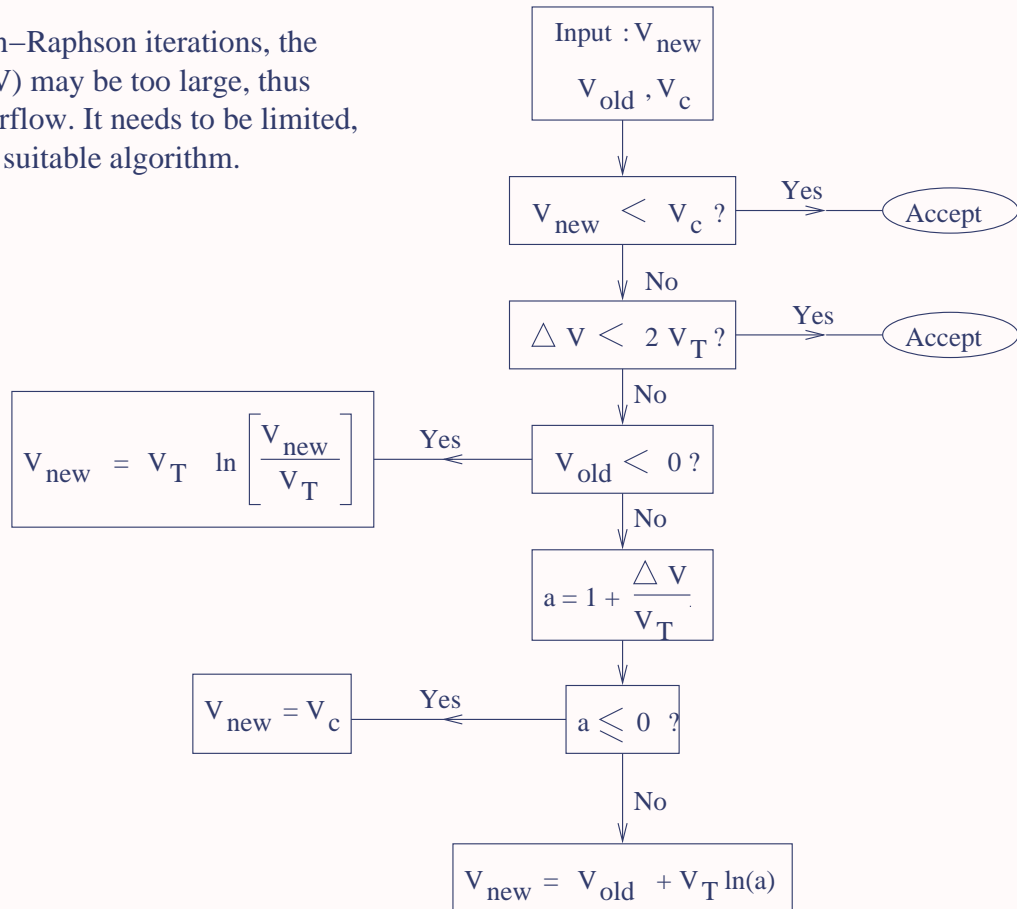
```
ebe name=c
nodes: p n
rparms: c=1.0
be_v: v1=p-n
stparms: v0sv=0
no_functions=1
f_1: d_dt(v1) current
fortran:
source:
  if (l_dc) then
    if (l_function) f(1)=cur(1)
    if (l_jacobian) then
      dfdv(1,1)=0.
      dfdi(1,1)=1.0
    end if
  end if
  if (l_trns) then
    c=rparm(1)
    if (l_function) f(1)=cur(1)/c
    if (l_jacobian) then
      dfdv(1,1)=0.
      dfdi(1,1)=1.0/c
    end if
  end if
  if (l_startup) then
    v0sv=stparm(1)
    if (l_function) f(1)=v(1)-v0sv
    if (l_jacobian) then
      dfdv(1,1)=1.
      dfdi(1,1)=0.
    end if
  end if
endfortran
endebe
```

Equations:

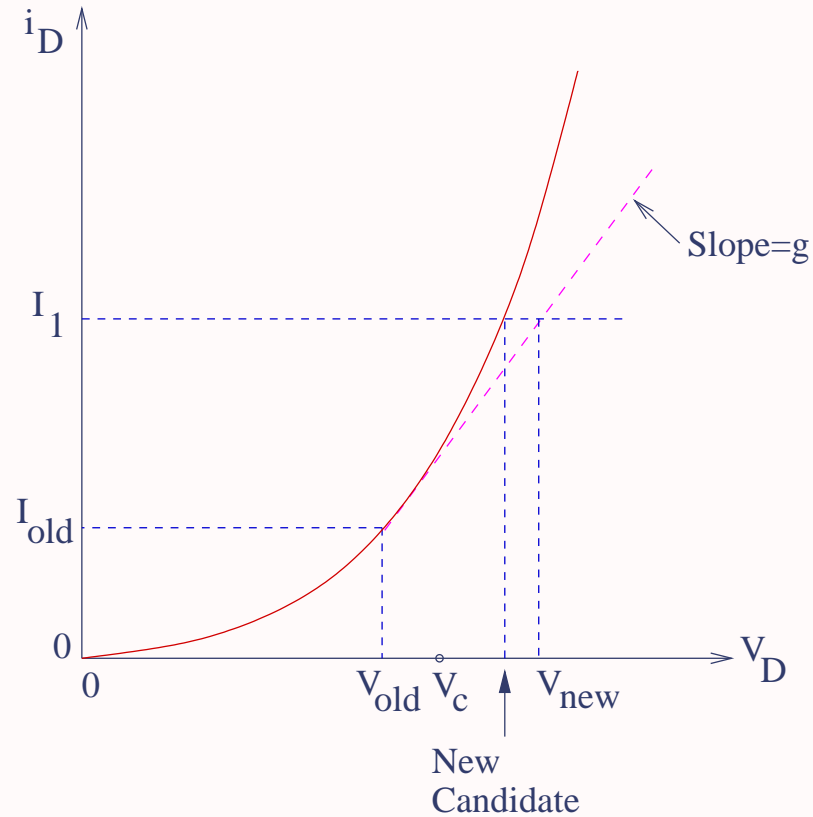
- DC: $i=0$
- Transient: $\frac{1}{C} i = \frac{dV}{dt}$
- “Start-up”: $V - V_0=0$

Limiting of diode voltage (SPICE)

During Newton–Raphson iterations, the correction (ΔV) may be too large, thus causing an overflow. It needs to be limited, by following a suitable algorithm.



Limiting of diode voltage (SPICE)



SEQUEL: diode template

```
ebe name=diode1 limit_newton=yes
nodes: p n
rparams: eta=1.0 is=1e-14 vt=0.026
+         eg=1.1 xti=3 is0=1.0e-14
be_v: v1=p-n
tmpr: t1
no_functions=1
f_1: v1 current
fortran:
variables:
    real*8 a1,a2,v1new,vcrit,v_limit,delta_v,arg
source:
    if (l_new_tmpr) then
        eta=rparm(1)
        t1=tmpr(1)
        vt=0.02585*t1/300.0
        etavt=eta*vt
        xti=rparm(8)
        eg=rparm(7)
        is0=rparm(9)
        is=is0*((t1/300.0)**xti)
        is=is*exp(eg*((1.0/0.02585)-(1.0/vt)))
        rparm(3)=vt
        rparm(4)=etavt
        rparm(2)=is
    end if
```

SEQUEL: diode template (continued)

```
if (l_dc.or.l_trns.or.l_startup) then
  is=rparm(2)
  etavt=rparm(4)
  v1=v(1)
  a1=exp(v1/etavt)
  if (l_function) f(1)=cur(1)-is*(a1-1.0)
  if (l_jacobian) then
    dfdi(1,1)=1.0
    dfdv(1,1)=-is*a1/etavt
  end if
end if
```

SEQUEL: diode template (continued)

```
if (l_limit_newton) then
  vlnew=v_new(1)
  vt=rparm(3)
  vcrit=vt*rparm(5)
  if (vlnew.gt.vcrit) then
    v_limit=vt+vt
    v1=v(1)
    delta_v=vlnew-v1
    if (abs(delta_v).gt.v_limit) then
      if (v1.lt.0.) then
        v_new(1)=vt*log(vlnew/vt)
      else
        arg=1.0+(delta_v/vt)
        if (arg.lt.0.) then
          v_new(1)=vcrit
        else
          v_new(1)=v1+vt*log(arg)
        end if
      end if
    end if
  end if
end if
endfortran
endebe
```

SEQUEL: LUT template (MOS transistor)

```
ebe name=lutn1
nodes: term1 term2 gate bulk
rparms: ratio_w=1.0 gmin=1.0e-12
be_v: v1=term1 v2=term2 vgate=gate vbulk=bulk
aux_var: qgate qbulk q2
stparms: device_on=0.0
no_functions=6
f_1: qgate[1] v1 v2 vgate vbulk
f_2: qbulk[1] v1 v2 vgate vbulk
f_3: q2[1] v1 v2 vgate vbulk
f_4: cur(gate)[1] d_dt(qgate)
f_5: cur(bulk)[1] d_dt(qbulk) vbulk v1 v2
f_6: cur(term2)[1] d_dt(q2) v1 v2 vgate vbulk
```

SEQUEL: LUT template (MOS transistor) continued

fortran:

variables:

```
integer nr_ratio_w,nr_gmin,nr_ron,nr_roff
data nr_ratio_w,nr_gmin,nr_ron,nr_roff/1,2,3,4/
integer nnd_term1,nnd_term2,nnd_gate,nnd_bulk
data nnd_term1,nnd_term2,nnd_gate,nnd_bulk/1,2,3,4/
integer nbev_v1,nbev_v2,nbev_vg,nbev_vb
data nbev_v1,nbev_v2,nbev_vg,nbev_vb/1,2,3,4/
integer na_qgate,na_qbulk,na_q2
data na_qgate,na_qbulk,na_q2/1,2,3/
integer nig_vlig,nig_v2ig,nig_vgig,nig_vbig
data nig_vlig,nig_v2ig,nig_vgig,nig_vbig/1,2,3,4/
integer nst_device_on
data nst_device_on/1/
integer neq_qg,neq_qb,neq_q2,neq_ig,neq_ib,neq_i2
data neq_qg,neq_qb,neq_q2,neq_ig,neq_ib,neq_i2/1,2,3,4,5,6/

real*8 vds,vgs,vbs,vsrc,vdrn
real*8 id_lut
real*8 qd_lut,qg_lut,qb_lut
real*8 id_vd_lut,id_vg_lut,id_vb_lut
real*8 qd_vd_lut,qd_vg_lut,qd_vb_lut
real*8 qg_vd_lut,qg_vg_lut,qg_vb_lut
real*8 qb_vd_lut,qb_vg_lut,qb_vb_lut
integer term_src,term_drn
```

SEQUEL: LUT template (MOS transistor) continued

source:

```
ratio_w=rparm(nr_ratio_w)
gmin=rparm(nr_gmin)

if (l_one_time_parms) then
  call lut_read_file_n
  call define_tau
  call lucoef_n
  return
end if

if (l_dc.or.l_trns) then
  v1=v(nbev_v1)
  v2=v(nbev_v2)
  vgate=v(nbev_vg)
  vbulk=v(nbev_vb)

  if (v2.ge.v1) then
    term_src=1
    term_drn=2
  else
    term_src=2
    term_drn=1
  end if
```

SEQUEL: LUT template (MOS transistor) continued

```
vsrc=v(term_src)
vdrn=v(term_drn)

vds=vdrn-vsrc
vgs=vgate-vsrc
vbs=vbulk-vsrc

call luinterp_n(
#   vds,vgs,vbs,ratio_w,
#   id_lut,qd_lut,qg_lut,qb_lut,
#   id_vd_lut,id_vg_lut,id_vb_lut,
#   qd_vd_lut,qd_vg_lut,qd_vb_lut,
#   qg_vd_lut,qg_vg_lut,qg_vb_lut,
#   qb_vd_lut,qb_vg_lut,qb_vb_lut)
```

SEQUEL: LUT template (MOS transistor) continued

```
if (l_function) then
  qgate=auxvar(na_qgate)
  qbulk=auxvar(na_qbulk)
  q2=auxvar(na_q2)

  f(neq_qg)=qgate-qg_lut
  f(neq_qb)=qbulk-qb_lut

  if (term_drn.eq.2) then
    f(neq_q2)=q2-qd_lut
  else
    f(neq_q2)=q2+(qd_lut+qg_lut+qb_lut)
  end if

  f(neq_ig)=cur(nnd_gate)
  f(neq_ib)=cur(nnd_bulk)+gmin*(v1+v2-2*vbulk)

  if (term_drn.eq.2) then
    f(neq_i2)=cur(nnd_term2)+gmin*(vbulk-v2)-id_lut
  else
    f(neq_i2)=cur(nnd_term2)+gmin*(vbulk-v2)+id_lut
  end if
end if
```

SEQUEL: LUT template (MOS transistor) continued

```
if (l_jacobian) then
  dqddvg=qd_vg_lut
  dqgdvg=qg_vg_lut
  dqbdvg=qb_vg_lut

  diddvg=id_vg_lut

  dqddvb=qd_vb_lut
  dqgdvb=qg_vb_lut
  dqbdvb=qb_vb_lut

  diddvb=id_vb_lut

  if (term_drn.eq.2) then
    dqddv1=-qd_vd_lut-qd_vg_lut-qd_vb_lut
    dqgdv1=-qg_vd_lut-qg_vg_lut-qg_vb_lut
    dqbdv1=-qb_vd_lut-qb_vg_lut-qb_vb_lut

    dqddv2= qd_vd_lut
    dqgdv2= qg_vd_lut
    dqbdv2= qb_vd_lut

    diddv1=-id_vd_lut-id_vg_lut-id_vb_lut
    diddv2= id_vd_lut
  (the other case has been dropped, it is similar)
end if
```

C

SEQUEL: LUT template (MOS transistor) continued

```
dfdv(neq_qg,nbev_v1)=-dqgdv1  
dfdv(neq_qg,nbev_v2)=-dqgdv2  
dfdv(neq_qg,nbev_vg)=-dqgdvg  
dfdv(neq_qg,nbev_vb)=-dqgdvb
```

```
dfdv(neq_qb,nbev_v1)=-dqbdv1  
dfdv(neq_qb,nbev_v2)=-dqbdv2  
dfdv(neq_qb,nbev_vg)=-dqbdvg  
dfdv(neq_qb,nbev_vb)=-dqbdvb
```

```
if (term_drn.eq.2) then  
    dfdv(neq_q2,nbev_v1)=-dqddv1  
    dfdv(neq_q2,nbev_v2)=-dqddv2  
    dfdv(neq_q2,nbev_vg)=-dqddvg  
    dfdv(neq_q2,nbev_vb)=-dqddvb  
end if
```

SEQUEL: LUT template (MOS transistor) continued

```
dfdaux(neq_ig,na_qgate)=0.0

dfdv(neq_ib,nnd_term1)= gmin
dfdv(neq_ib,nnd_term2)= gmin
dfdv(neq_ib,nnd_bulk)=-2*gmin
dfdaux(neq_ib,na_qbulk)=0.0

if (term_drn.eq.2) then
    dfdv(neq_i2,nnd_term1)=-diddv1
    dfdv(neq_i2,nnd_term2)=-diddv2-gmin
    dfdv(neq_i2,nnd_gate)= -diddvg
    dfdv(neq_i2,nnd_bulk)= -diddvb+gmin
end if
dfdaux(neq_i2,na_q2)=0.0
end if
end if
endfortran
endebe
```

References:

- (1) B. R. Chawla, H. K. Gummel, and P. Kozak, "MOTIS-An MOS timing simulator," *IEEE Trans. Circuits Syst.*, vol. CAS-22, no. 12, pp. 901-910, Dec. 1975.
- (2) A. R. Newton, "Techniques for the simulation of large-scale integrated circuits," *IEEE Trans. Circuits and Syst.*, vol. CAS-26, pp. 741-749, Sept. 1979.
- (3) T. Shima, T. Suguwara, S. Moriyama, and H. Yamada, "Three-dimensional table look-up MOSFET model for precise circuit simulation," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 3, pp. 449-453, June 1982.
- (4) W. Coughran, E. Grosse, and D. Rose, "CAzM: A circuit analyzer with macromodeling," *IEEE Trans. Electron Devices*, vol. ED-30, pp. 1207-1213, Sept. 1983.
- (5) J. L. Huertas and A. Rueeda, "Sectionwise piecewise polynomial functions: Applications to the analysis and synthesis of nonlinear n-port networks," *IEEE Trans. Circuits Syst.*, vol. CAS-31, pp. 897-906, Oct. 1984.
- (6) G. Bischoff and J. P. Krusis, "Technology independent device modeling for simulation of integrated circuits for FET technologies," *IEEE Trans. Computer-Aided Design*, vol. CAD-4, no. 1, pp. 99-110, Jan. 1985.
- (7) L. O. Chua and A. Deng, "Canonical piecewise linear modeling," *IEEE Trans. Circuits Syst.*, vol. CAS-33, no. 5, pp. 511-525, May 1986.
- (8) P. B. L. Meijer, "Fast and smooth highly nonlinear multidimensional table models for device modeling," *IEEE Trans. Circuits and Systems*, vol. 37, pp. 335-345, March 1990.
- (9) Y.-H. Shih and S. M. Kang, "ILLIADS: A New Fast MOS Timing Simulator Using Direct Equation-Solving Approach", *DAC 1991*, pp 20-25.

References (continued):

- (10) M. Yanilmaz and V. Eveleigh, "Numerical device modeling for electronic circuit simulation," *IEEE Trans. Computer-Aided Design*, vol. 10, pp. 366-375, May 1991.
- (11) C. Visweswariah and R. A. Roher, "Piecewise approximate circuit simulation," *IEEE Trans. Computer-Aided Design*, pp. 861-890, July 1991.
- (12) D. H. Cho and S. M. Kang, "An Accurate AC Characteristic Table Look-up Model for VLSI Analog Circuit Simulation Applications," *IEEE Int. Symp. on Circuits and Syst.*, Chicago, IL, May 1993, pp. 1531-1534.
- (13) M. G. Graham, J. J. Paulos, and D. W. Nychka, "Template-based MOSFET device model," *IEEE Trans. Computer-Aided Design Integrated Circuits and Systems*, vol. 14, no. 8, pp. 924-933, August 1995.
- (14) F. Dartu and L. T. Pileggi, "TETA: Transistor-level Engine for Timing Analysis," *Proc. 35th Design Automation Conf.*, 1998.
- (15) E. P. Vandamme, D. Shreurs, C. Van Dinther, G. Badenes, and L. Deferm, "Development of a RF large signal MOSFET model based on an equivalent circuit, and comparison with the BSIM3v3 compact model," *Solid-State Electron.*, vol. 46, no. 3, pp. 353-360, March 2002.

Thank you!