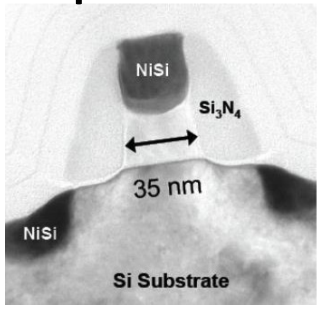
COMPACT MODELING USING VERILOG-A AND INTEGRATION WITH HSPICE

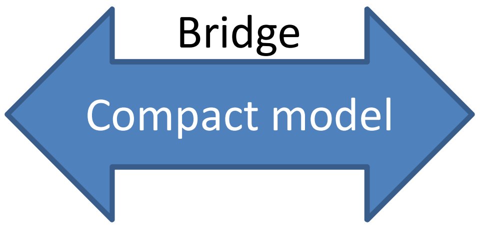
What is a compact model?

-> Computationally efficient description of the terminal properties of a device as a function of terminal voltages

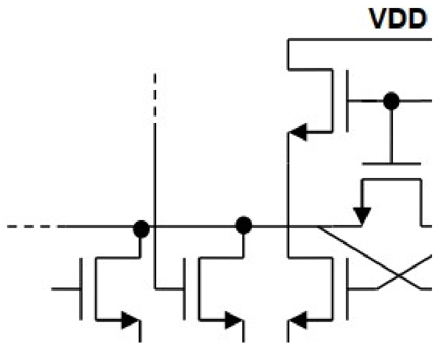
[{I}, {Q}] = f(Vg, Vd, Vs, Vb)

PROCESS/TECHNOLOGY Development





CIRCUIT Designing



-> The compact model is implemented inside a circuit simulation engine.
-> How to make a compact model?

Different types of Compact models:

▪ **Physics based**

All parameters have physical significance.
Computationally efficient and technology independent.

▪ **Empirical**

Set of equations with fitting parameters.

▪ **Lookup table based**

Limited to characterized data present in lookup table.

▪ **Macromodels**

Present in the form of equivalent circuit representation.

▪ **Semi-empirical model**

Introduced fitting parameters in Physics based model to capture device behaviour.
Empirical lookup table based model.

Compact model Requirements:

▪ **Model stability and convergence**

Functions and their derivatives must be continuous
No evaluation resulting 0/0 (i.e. physical limit must be accurately set)

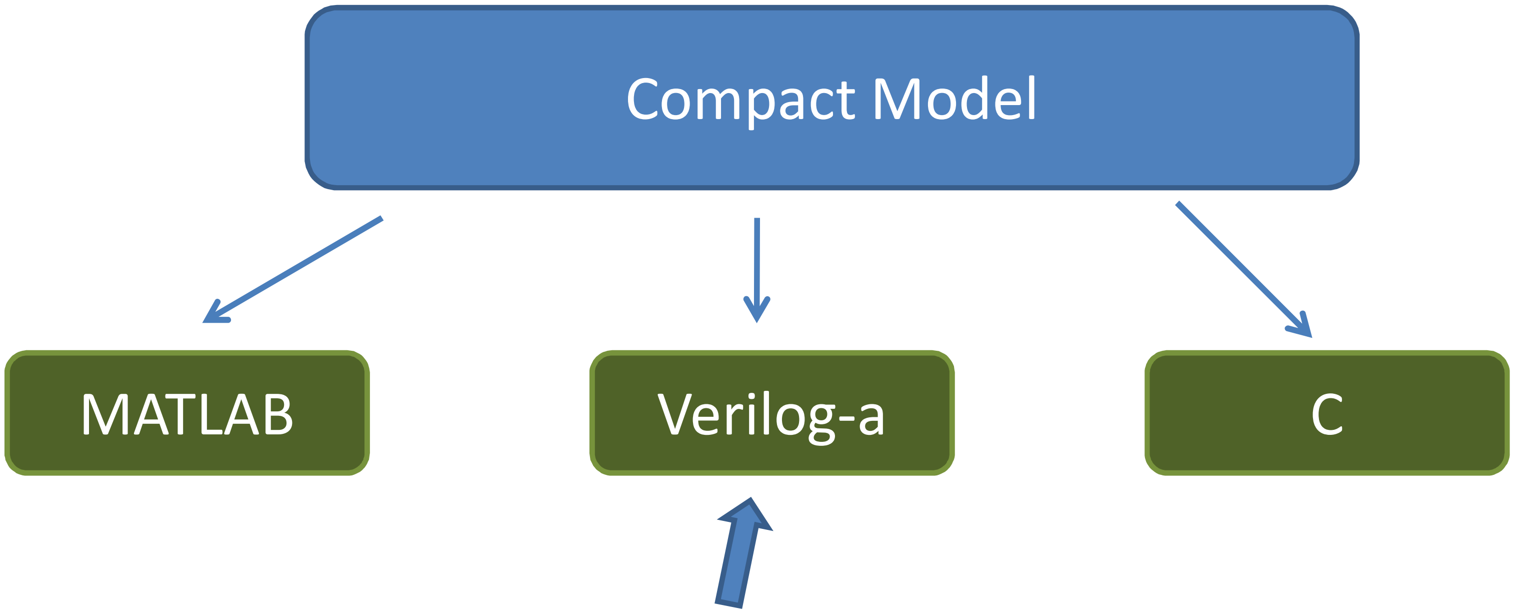
▪ **Speed of evaluation**

Expensive math functions must be avoided.
Reusability of internal variables to reduce computation time.

▪ **Accuracy**

Must be accurate up to required RMS error.

Ref: Tutorial on “Basics of Compact Model Development”, by Sivakumar P Mudanai (*Intel Corporation, Santa Clara, CA*).
Web: https://nanohub.org/resources/21367/



‘Hardware Description Language’
Can be easily integrated with SPICE-Simulators

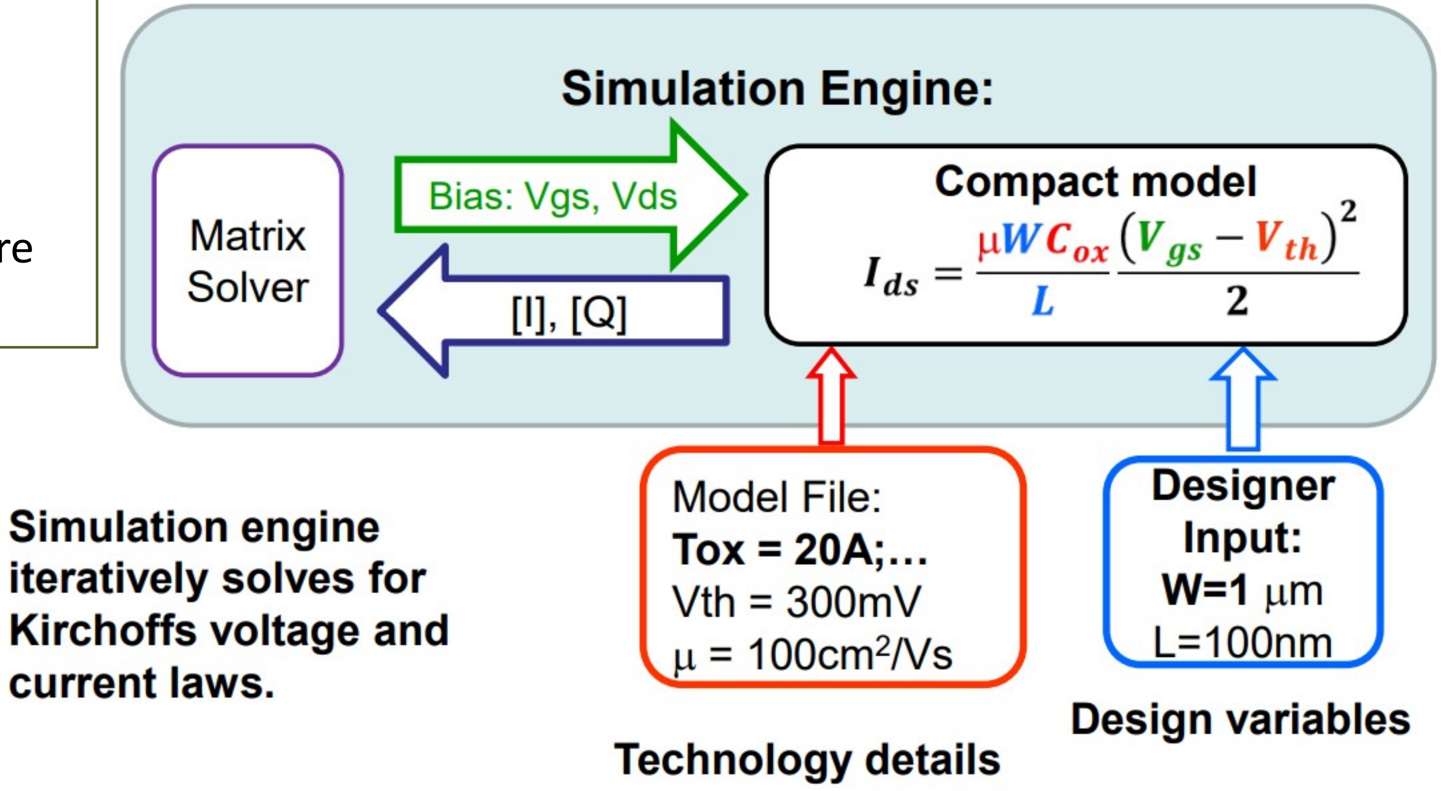
https://nanohub.org/resources/20580/download/2014.02.21-Coram-NEEDS.pdf

Circuit Simulators??

- ❑ NgSpice
- ❑ HSpice
- ❑ PSpice
- ❑ MultiSim
- ❑ CadenceSpectre

…..

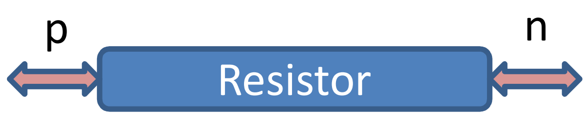
SIMPLIFIED VIEW OF A CIRCUIT SIMULATION



Verilog-A Model for a Simple Resistor

Save the following code as Resistor.va (can use any text editor)

```
`include "disciplines.vams"
module simple_resistor(p,n) ;
inout p,n;
parameter real resistance = 1000.0 from (0.0:inf) ;
electrical p, n ;
analog
begin
I(p,n) <+ V(p,n)/resistance ;
end
endmodule
```

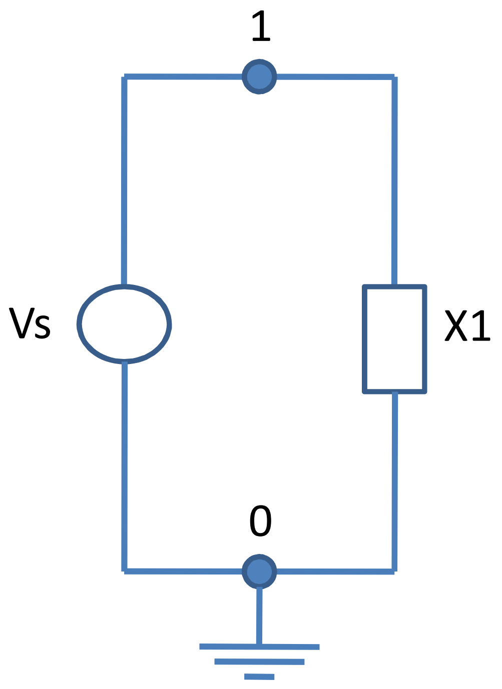


First line includes header files which define "electrical" discipline and access functions V and I (also thermal, kinematic, …)

https://nanohub.org/resources/20580/download/2014.02.21-Coram-NEEDS.pdf

Hspice file for a Simple Resistor

```
* Basic Resistor
******************************
* Include Files (* means comment)
******************************
.OPTION POST=2 INGOLD=2 $ (text after $ is comment)
.hdl “Resistor.va“ $ calling the verilog-a file
.model res simple_resistor $ creating a model instance
******************************
* Netlist
******************************
X1 1 0 res resistance=100 $ instantiating the instance as sub-circuit
Vs 1 0 DC 0.8
******************************
.dc Vs 0 4 0.2
.print V(1)
.end
```



Save this code as Resistor.sp

https://cseweb.ucsd.edu/classes/wi10/cse241a/assign/hspice_sa.pdf

Verilog-A model of a Capacitor

Save the following code as Capacitor.va

```
`include "disciplines.vams"
module simple_capacitor(p,n);
inout p,n;
electrical p,n;
parameter real C=1e-6 from [0.0:inf); //default value
real q;
analog begin
q=C*V(p,n);
I(p,n)<+ ddt(q);
end
endmodule
```

Q1. Use your customized models for resistor and capacitor to design RC-Low Pass Filter with 3-dB Bandwidth of 1KHz. Perform AC analysis to verify the result. Overwrite appropriate values in place of default values for Resistor and Capacitor. Draw the circuit showing Vs, X1, X2, and node labels.

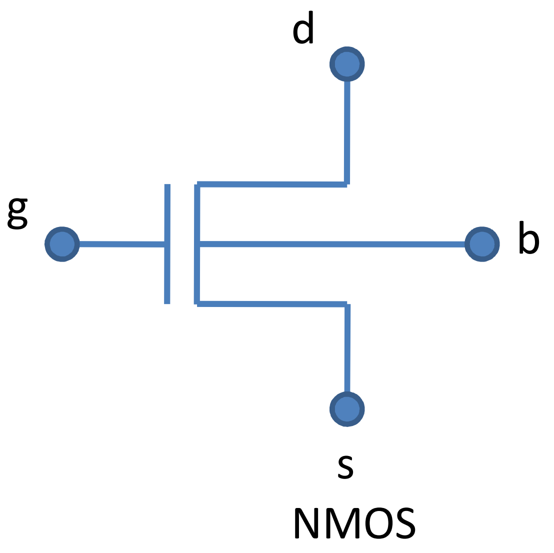
```
*RC-Check
********Include Files*******
.OPTION POST=2
.hdl “Resistor.va"
.hdl “Capacitor.va"
.model cap simple_capacitor
.model res simple_resistor
******* Netlist ************
X1 1 2 res resistance=What value?
X2 2 0 cap C=What value?
Vs 1 0 AC 1V
************************
.AC DEC 10 1 1MEG
.print V(2)
.end
```

Save this code as RC.sp

Q2(a). Make a Verilog-A model for a simple n-channel MOSFET (NMOS) by defining current equations in linear and saturation regions. Verify by plotting INPUT and OUTPUT characteristics in HSPICE. Consider Vth=0.6V.

```
`include "disciplines.vams"
`include "constants.vams"
module simple_NMOS(d,g,s,b);
inout d,g,s,b; // inout TERMINALS
electrical d ,g ,s ,b ; //input-output nodes
parameter real W=1e-5 from [0:inf]; // width of MOSFET
parameter real L=1e-5 from [0:inf]; // Length of MOSFET
parameter real Vth=0.6 from [0:inf]; //  Threshold voltage
parameter real mu=1400 from [0:inf];
parameter real NA=1e17 from [1e13:1e22]; // Doping
parameter real TOX=20e-7; // Oxide Thickness
parameter real VSB=0.0; //Body-Bias
real P_EPSOX=3.9*8.85e-14;
real P_EPSSI=11.7*8.85e-14;
real PHI,GAMMA,COX,PHI_F,VT; // You can choose your own variables
real Ni=1e10;
real q=1.6e-19;
analog
begin
// ……. Real variables calculations using the provided equations………
if (V(g,s) > Vth) //ensuring conduction after threshold
begin
if(……)
……
else
……
end
end
endmodule
```

Save this code as NMOS.va



Reference Hspice code for NMOS

```
* Basic mos_test
******** Include Files
.OPTION POST=2 INGOLD=2
.hdl “MOSFET.va" $ change the names according to your file_names
.model mos simple_mos $ change module name accordingly
****NETLIST
X1 3 2 0 0 mos
Vdum 4 3 DC 0
$R 4 3 100K $ Uncomment for INVERTER problem
Vdd 4 0 DC 2
Vgg 2 0 DC 2
$ Vgg 2 0 PULSE(0 2.5 5n 2.5n 2.5n 20n 45n) $ uncomment for INVERTER problem
*****Analysis $ LV HV td tr tf PW PER
.dc Vdd 0 2 0.02 $ to plot Output Characteristics
.dc Vgg 0 2 0.02 $ to plot Input Characteristics
* .tran 250p 145n $ uncomment for INVERTER problem
$ .print – to get data in the .lis file
.end
```

Save this code as NMOS.sp

Q2(b). Design a basic INVERTER using the NMOS and verify by applying a square pulse at Gate.

Q3. Repeat Q2(a) for a PMOS transistor. Plot magnitude of current versus Vds (Vdd=0 to -2V) at Vgg=-2V, and magnitude of current versus Vgs (Vgg=0 to -2V) at Vdd=-2V. Consider Vth=-0.6V.
Note the following differences compared to NMOS: PMOS is ON when Vgs < Vth, is in linear region when Vds >= (Vgs-Vth), and is in saturation when Vds < (Vgs-Vth).

Q4. Design a CMOS inverter using NMOS (from Q2) and PMOS (from Q3) and verify by applying the same input square pulse as in Q2(b), however the pulse oscillates between 0V and 2V in this case. The circuit for CMOS is shown in Fig. A.

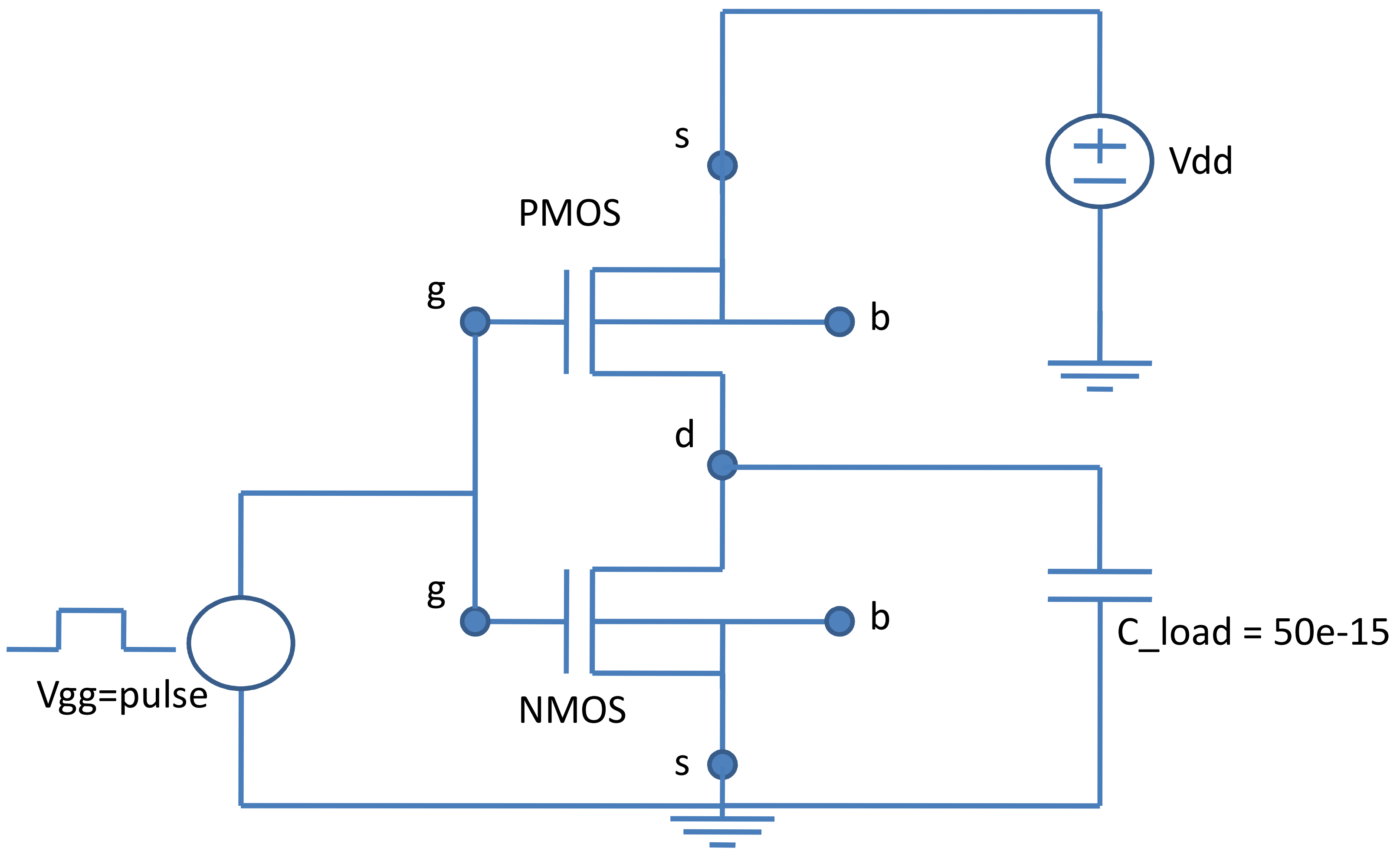


Fig. A: CMOS Inverter circuit