

---

# DISPENSE PER SIMULAZIONI MEDIANTE SOFTWARE SIMETRIX

---

Corso di Devices and Circuits with High Energy Efficiency for IOT Applications

# Sommario

1	Introduzione .....	4
1.1	Il modello .....	4
1.2	Progetto di simulazione .....	6
1.2.1	I file schematici.....	6
1.2.2	Gli script file .....	6
1.2.3	Command Shell .....	7
1.2.4	Gruppi.....	8
2	Modelli e sottocircuiti .....	11
2.1	Creazione di una libreria.....	11
2.2	Procedura per il caricamento di una libreria o un modello in SIMETRIX .....	12
2.3	Procedura per la creazione di sotto-circuiti.....	14
2.4	Utilizzo dei circuiti “compatti” .....	16
3	Progetto 1: Capacità di gate.....	17
3.1	Parametri di simulazione.....	17
3.2	Parametri del modello.....	17
3.3	Circuito e simulazione.....	17
3.4	Script file.....	18
4	Progetto 2: Caratteristica I/O di invertitore.....	19
4.1	Parametri di simulazione.....	19
4.2	Parametri del modello.....	19
4.3	Circuito e simulazione.....	19
4.4	Script file.....	20



# 1 Introduzione

## 1.1 Il modello

Il seguente documento si basa sul modello semi-empirico che esprime la corrente  $I_{DS}$  di un transistor come:

NMOS

$$I_{DS} = W Q_{SUB} F_{SD} v_{th} = W I_0 \ln \left( 1 + \exp \left( \frac{V_{gs} - V_t}{n \phi_t} \right) \right) \left( 1 - \exp \left( - \frac{V_{DS}}{\phi_t} \right) \right)$$

PMOS

$$I_{DS} = W Q_{SUB} F_{DS} v_{th} = W I_0 \ln \left( 1 + \exp \left( - \frac{V_{gs} - V_t}{n \phi_t} \right) \right) \left( 1 - \exp \left( \frac{V_{DS}}{\phi_t} \right) \right)$$

In entrambi i casi  $V_t = V_{t0p} - \lambda_{DS} V_{DS} + \lambda_{SB} V_{SB}$

Mentre  $I_0 = \frac{t_{CH}}{2 - t_{CH}} C_{inv} n \phi_t v_{th}$

Che codificato in formato SPICE diventa:

NMOS	PMOS
<pre>***** *UNIVERSITA' DEGLI STUDI DI UDINE *Students: BERTONI LUCA, RICCARDO FONTANINI *Supervisor: Prof. ESSENI DAVID *Assistant Supervisor: Dott. Ing. ROLLO TOMMASO *MODELLO NMOS *14/02/2019 *****#PARAMETERS#***** *dimensions.....W,L=[m]; *thermal voltage....Vth=[V]; *non-ideal factor...n=[a.u.]; *thermal velocity...Term_V=[m/s]; *oxide cap.....Cinv=[F/m2]; ***** .subckt SUB_NMOS drainin gatein sourcein .param T = 300 .param W = 1e-6 .param L = 30e-9 .param Vt0 = 0.35 .param n=1 .param tch = 0.70 .param Term_V = 1.23e5 .param Cinv = 0.020 .param Kds = 0.04 .param rpara = 60 .param Vt0n = 0.35 .param Vt0p = -0.35 *.param Vt0 = {{(Vt0n)}} .param Vth= {BOLTZ * {{(T)}} / ECHARGE} .param Vt= {{(Vt0)}}-{{(Kds)}}*V(drain,source)} .param I0= {{{(Cinv)}} * {{(n)}} * Vth * Term_V * ( {{(tch)}} / (2 - {{(tch)}})) } bswitch drain source I= {{{(W)}} * I0 * LN(1+exp( (V(gate,source)- Vt) / ( {{(n)}} * Vth ))) * (1-exp( -( V(drain,source))/ Vth ))) } *Parasitic drain, source, and gate resistances* rd drainin drain {{(rpara)}} rs sourcein source {{(rpara)}} rg gatein gate {{(.6)}} * Cg capacitance* C_GD gate drain {1e-16 * {{(W)}} * 1e6} C_GS gate source {1e-16 * {{(W)}} * 1e6} BC_GS gatein sourcein Q = {0.5 * {{(Cinv)}} * {{(W)}} * {{(L)}} * {{(n)}} * Vth * LN(1 + exp((V(gate,source) - Vt) / ({{(n)}} * Vth )))}} BC_GD gatein drainin Q={0.5 * {{(Cinv)}} * {{(W)}} * {{(L)}} * {{(n)}} * Vth * LN(1+exp((V(gate,drain)-Vt)/( {{(n)}} * Vth )))}} .ends</pre>	<pre>***** *UNIVERSITA' DEGLI STUDI DI UDINE *Students: BERTONI LUCA, RICCARDO FONTANINI *Supervisor: Prof. ESSENI DAVID *Assistant Supervisor: Dott. Ing. ROLLO TOMMASO *MODELLO PMOS *14/02/2019 *****#PARAMETERS#***** *dimensions.....W,L=[m]; *thermal voltage....Vth=[V]; *non-ideal factor...n=[a.u.]; *thermal velocity...Term_V=[m/s]; *oxide cap.....Cinv=[F/m2]; ***** .subckt SUB_PMOS drainin gatein sourcein .param T = 300 .param W = 1e-6 .param L = 30e-9 .param Vt0 = -0.35 .param n=1 .param tch = 0.70 .param Term_V = 1.23e5 .param Cinv = 0.020 .param Kds = 0.04 .param rpara = 60 .param Vt0n = 0.35 .param Vt0p = -0.35 *.param Vt0 = {{(Vt0p)}} .param Vth= {BOLTZ * {{(T)}} / ECHARGE } .param Vt= { {{(Vt0)}} - ({{(Kds)}} * V(drain,source)) } .param I0= {{{(Cinv)}} * {{(n)}} * Vth * Term_V * ( {{(tch)}} / (2 - {{(tch)}})) } bswitch drain source I= {-( {{(W)}} * I0 * LN(1+exp(- V(gate,source)-Vt) / ( {{(n)}} * Vth ))) * (1-exp( ( V(drain, source)) / Vth ))) } *Parasitic drain, source, and gate resistances* rd drainin drain {{(rpara)}} rs sourcein source {{(rpara)}} rg gatein gate {{(.6)}} * Cg capacitance* C_GD gate drain {1e-16 * {{(W)}} * 1e6} C_GS gate source {1e-16 * {{(W)}} * 1e6} BC_GS gatein sourcein Q = { 0.5 * {{(W)}} * {{(L)}} * {{(Cinv)}} * {{(n)}} * Vth * LN( 1 + exp(-(V(source,gate)-Vt) / ({{(n)}}*Vth)))}} BC_GD gatein drainin Q={0.5 * {{(W)}} * {{(L)}} * {{(Cinv)}} * {{(n)}} * Vth * LN(1+exp(-(V(drain,gate)-Vt) / ( {{(n)}}*Vth)))}} .ends</pre>

Al fine di utilizzare tale modello col software Simetrix<sup>1</sup> è utile parametrizzare il file descrittore, ed applicare i parametri dei transistori direttamente nell'ambiente simulativo. Questo permette di velocizzare le operazioni di creazione del circuito da simulare e migliorare l'elasticità dei modelli implementati.

I valori standard presi come riferimento sono:

T [K]	W [m]	L [m]	Vt0n [V]	Vt0p [V]	n	tch	Term_V [m/s]	Cinv [F/m <sup>2</sup> ]	Kds	rpara [Ω]
300	1e-6	30e-9	0,35	-0,35	1	0,7	1,23e5	0,02	0,04	60

Per utilizzare un modello parametrico è necessario modificare i due modelli visti precedentemente come segue:

PMOS:

```
.subckt SUB_PMOS drainin gatein sourcein
.param T = 300
.param W = 1e-6
.param L = 30e-9
.param Vt0 = -0.35
.param n=1
.param tch = 0.70
.param Term_V = 1.23e5
.param Cinv = 0.020
.param Kds = 0.04
.param rpara = 60
.param Vt0n = 0.35
.param Vt0p = -0.35
*.param Vt0 = {{(Vt0p)}}
.param Vth= {BOLTZ * {{(T)}} / ECHARGE }
.param Vt= { {{(Vt0)}} - ({{(Kds)}} * V(drain,source)) }
[...]
```

```
.subckt SUB_PMOS drainin gatein sourcein
*.param T = 300
*.param W = 1e-6
*.param L = 30e-9
*.param Vt0 = -0.35
*.param n=1
*.param tch = 0.70
*.param Term_V = 1.23e5
*.param Cinv = 0.020
*.param Kds = 0.04
*.param rpara = 60
*.param Vt0n = 0.35
*.param Vt0p = -0.35
*.param Vt0 = {{(Vt0p)}}
.param Vth= {BOLTZ * {{(T)}} / ECHARGE }
.param Vt= { {{(Vt0)}} - ({{(Kds)}} * V(drain,source)) }
[...]
```

NMOS:

```
.subckt SUB_NMOS drainin gatein sourcein
.param T = 300
.param W = 1e-6
.param L = 30e-9
.param Vt0 = 0.35
.param n=1
.param tch = 0.70
.param Term_V = 1.23e5
.param Cinv = 0.020
.param Kds = 0.04
.param rpara = 60
.param Vt0n = 0.35
.param Vt0p = -0.35
*.param Vt0 = {{(Vt0n)}}
.param Vth= {BOLTZ * {{(T)}} / ECHARGE}
.param Vt= {{(Vt0)}}-{{(Kds)}}*V(drain,source)}
[...]
```

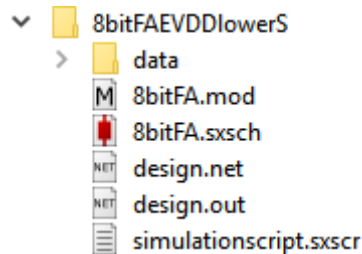
```
.subckt SUB_NMOS drainin gatein sourcein
*.param T = 300
*.param W = 1e-6
*.param L = 30e-9
*.param Vt0 = 0.35
*.param n=1
*.param tch = 0.70
*.param Term_V = 1.23e5
*.param Cinv = 0.020
*.param Kds = 0.04
*.param rpara = 60
*.param Vt0n = 0.35
*.param Vt0p = -0.35
*.param Vt0 = {{(Vt0n)}}
.param Vth= {BOLTZ * {{(T)}} / ECHARGE}
.param Vt= {{(Vt0)}}-{{(Kds)}}*V(drain,source)}
[...]
```

Commentando i parametri è quindi possibile sfruttare le variabili globali, definite all'interno dell'ambiente SIMetrix, per modificare a piacimento il comportamento dei dispositivi simulati..

<sup>1</sup> <https://www.simetrix.co.uk/>

## 1.2 Progetto di simulazione

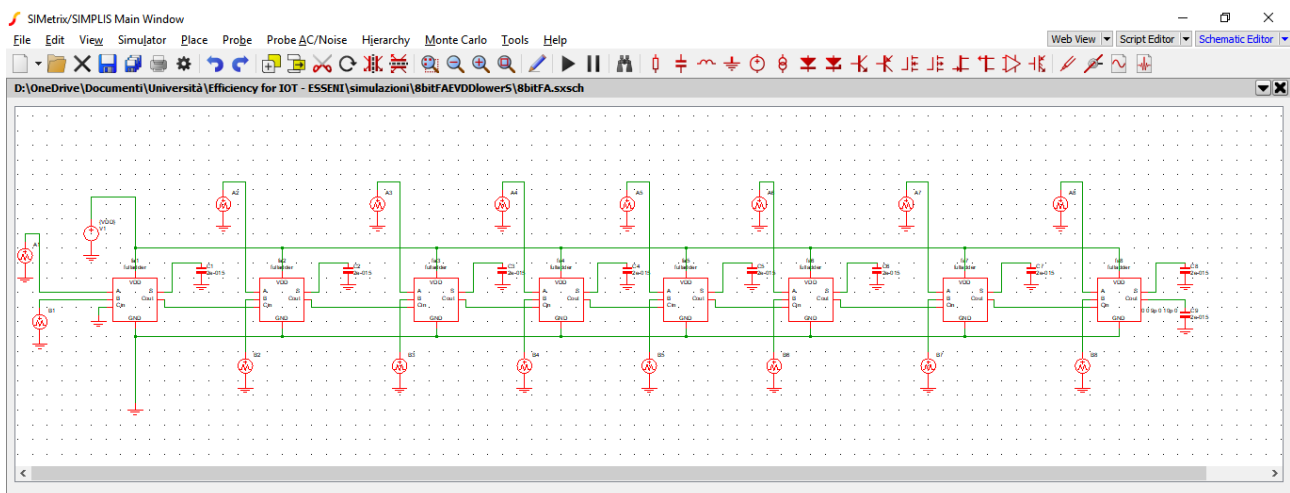
Ogni progetto di simulazione deve definire la caratteristica da simulare. Ad ogni progetto deve essere assegnata una cartella con un nome specifico. All'interno di essa devono essere presenti diversi file, i cui principali sono:



- un file \*.sxsch che rappresenta la topologia del circuito da simulare
- un file \*.sxsch contenente uno script SIMetrix che definisce modalità, tipologie di simulazione nonché i valori dei componenti utilizzati nei circuiti ed i parametri di modello.
- Una cartella data dove sono salvati i dati prodotti da varie simulazioni, che in certi casi possono essere onerose dal punto di vista tempistici.

### 1.2.1 I file schematici

È utile usare gli schematici solamente per definire la topologia del circuito, senza che vengano imposte grandezze da misurare direttamente sullo schematico (non vengono utilizzati i probes). Questo perché le grandezze da misurare possono essere definite dinamicamente nello script permettendo di raggruppare i dati in modo da poterli elaborare sequenzialmente in maniera più efficiente ed automatica.



In questa fase, cioè quando si crea la topologia del progetto, non è necessario definire i valori dei dispositivi inseriti nello schematico, infatti verranno anch'essi definiti dinamicamente dallo script file.

### 1.2.2 Gli script file

SIMetrix permette l'utilizzo di particolare Script (scritti in un linguaggio simile al BASIC) per automatizzare diverse operazioni. I file di scripting sono utilizzati ampiamente per definire dinamicamente parametri simulativi, tipologie di simulazioni e tutte le informazioni necessarie per la simulazione. I cicli vengono utilizzati per eseguire simulazioni variando parametri ottenuti da simulazioni precedenti. Per quanto

riguarda la sintassi si rimanda alle pagine<sup>2</sup> dedicate. Ogni progetto deve avere un suo script dedicato che varia in base alle operazioni da svolgere. In ogni caso la struttura di base rimane sempre la stessa:

1. Definizione e applicazione dei parametri sul circuito da simulare;

```
75 Unselect
76 Select /prop Ref C1
77 Prop value {stdCap}
```

2. Generazione della netlist e simulazione

```
173
174 Netlist design.net
175 Run /noerr /file design.net /an {' .TRAN 0 '& STR(propagationtimeinterval) '& 0 '& STR(maxstep)}
176
```

3. Elaborazione dei dati di simulazione

```
291
292 let supplycurrent = V1#n
293 let powerSupply = supplycurrent * VDDs[idVDD]
294 let energyHigh = integ (powerSupply)
295
```

4. Visualizzazione dei risultati

```
311
312 for energyindex = 0 to NEnergy-1
313 let energy = vector(NVDD)
314 for i=0 to NVDD-1
315 let energy[i] = totalenergies[energyindex * NVDD + i]
316 next NVDD
317 if energyindex == 0 then
318 Plot XY(energy, VDDs) /name {'Energy curve ' & str(energyindex)} /ylabel {'Energy / OP'} /unit {'J'}
319 else
320 Curve XY(energy, VDDs) /name {'Energy curve ' & str(energyindex)}
321 endif
322 next energyindex
```

Ogni volta che si lancia una simulazione, SIMetrix crea un Gruppo dove salvare, per ogni step simulativo, i valori di ogni nodo della rete. In questo modo è possibile lanciare diverse simulazioni salvando di volta in volta l'identificativo del Gruppo creato per tale simulazione. L'elaborazione e la visualizzazione dei dati di una determinata simulazione possono quindi essere eseguite a posteriori, richiamando l'identificativo del Gruppo che si vuole elaborare.

L'impostazione così definita facilita l'utilizzo degli ambienti simulativi, che altrimenti potrebbero risultare piuttosto ostici nelle fasi iniziali del loro impiego. Inoltre, essendo il codice intrinsecamente sequenziale, facilita molto la lettura delle operazioni svolte e la replicabilità di simulazioni complesse.

Gli script file devono essere lanciati mediante il tasto “Run Script”.

Il manuale utente di SIMetrix<sup>3,4</sup> contiene una descrizione approfondita dei comandi e del sistema di sviluppo.

### 1.2.3 Command Shell

La command shell di Simetrix permette di visualizzare parametri simulativi e di eseguire tutti i comandi compatibili con gli script file. In particolare è possibile visualizzare “run-time” tutte le grandezze derivate da una simulazione con il comando “Display”

<sup>2</sup>[https://help.simetrix.co.uk/8.0/simetrix/simetrix\\_docs.htm#mergedProjects/script\\_manual/topics/sr\\_com\\_curve.htm](https://help.simetrix.co.uk/8.0/simetrix/simetrix_docs.htm#mergedProjects/script_manual/topics/sr_com_curve.htm)

<sup>3</sup> <https://www.simetrix.co.uk/Files/manuals/8.2/UsersManual.pdf>

<sup>4</sup> <https://www.simetrix.co.uk/documentation/manuals.html>

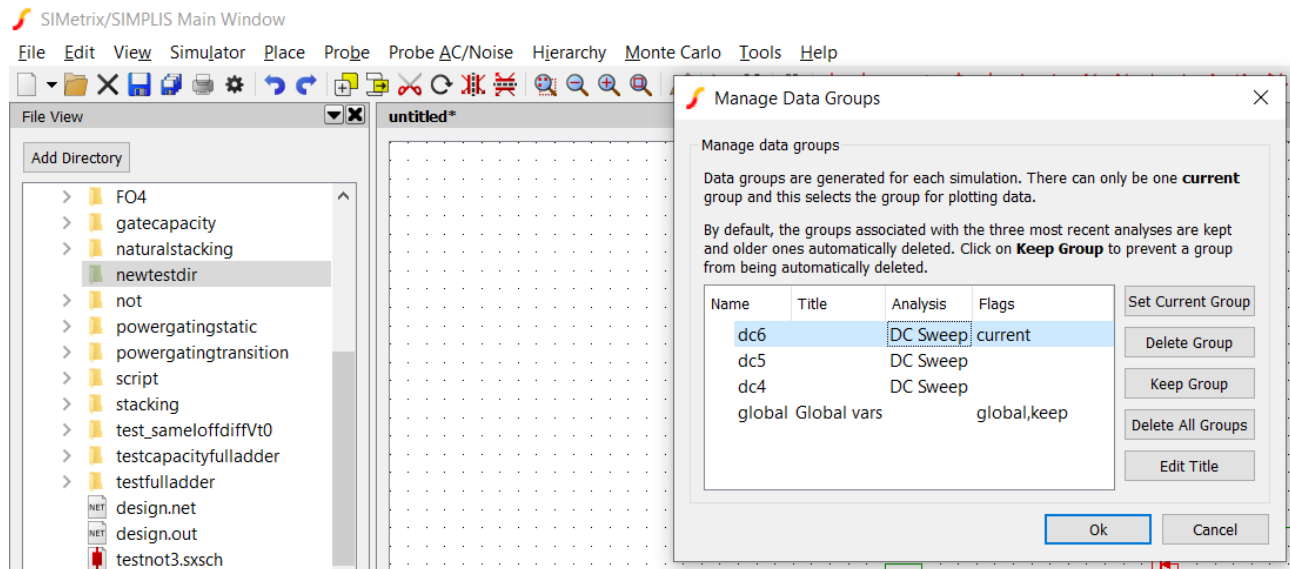




SIMetrix mette a disposizione un “gestore di gruppi” con il quale accedere, attraverso una finestra dedicata, ai vari gruppi dati presenti in memoria. In generale nelle impostazioni di SIMetrix esiste un campo che definisce il numero massimo di gruppi da mantenere in memoria. In questa scheda un particolare gruppo può essere mantenuto un gruppo in memoria, nonostante il numero di gruppi sia superiore a tale valore, tale impostazione è definita come “Keep Group”.

Per accedere a tale GUI:

### Simulator > Manage Data Groups..



Nel caso in cui vengano utilizzati gli script per eseguire una o più simulazioni, è importante utilizzare in modo appropriato i gruppi dati. Ipotizzando di voler variare un parametro del modello che si sta utilizzando per un numero  $n$  finito di volte al fine di ottenere le caratteristiche relative a  $n$  simulazioni differenti, è possibile utilizzare i gruppi per visualizzare i risultati di tale computazione. Inoltre, per ogni gruppo, è necessario impostare l'opzione KeepGroup via codice per non rischiare di perdere i dati durante le varie simulazioni. Un semplice esempio è rappresentato dal seguente codice:

```

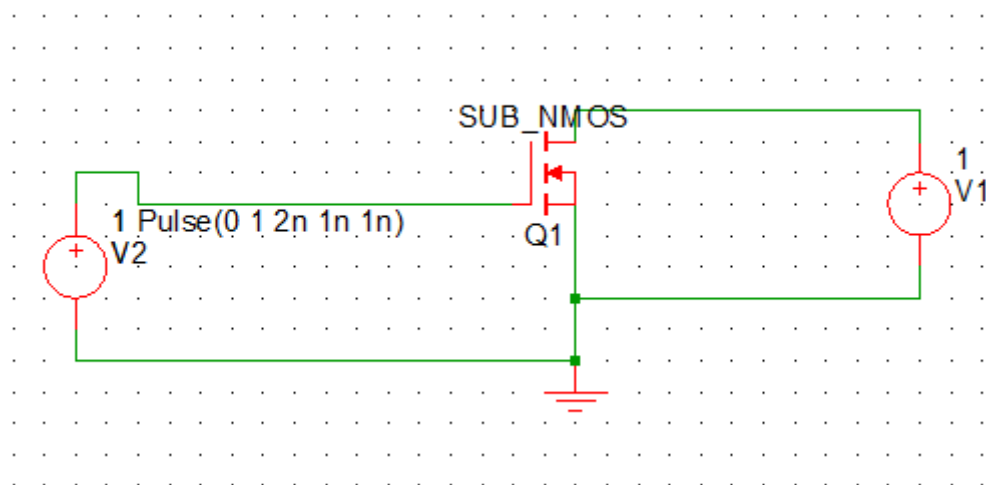
1 ClearMessageWindow
2 DelGroup /all
3 close Graph
4
5
6 let groupvector          = ['', '', '', '', '']
7 let N                    = 5
8 let global:T              = 300
9 let beckT                 = T
10
11 for i = 0 to N-1
12     * PARAMETER VARIATION *
13     let global:T          = beckT + (20 * i)
14     * SIMULATION *
15     Netlist design.net
16     Run /noerr /file design.net /an {'.TRAN 0 10n 0 30p'}
17     let groupvector[i]    = (Groups())[0]
18     KeepGroup {groupvector[i]}
19 next i
20
21 * RESTORE VARIABLE
22 let global:T              = beckT
23
24 * VISUALIZATION *
25 for i = 0 to N-1
26     SetGroup {groupvector[i]}
27     if i == 0 then
28         Plot Q1#drainin /ylabel "Current" /yunit "A" /name {'T = ' & Str(T)}
29     else
30         Curve Q1#drainin /name {'T = ' & Str(T + (20 * i))}
31     endif
32 next i
33
34 exit all

```

In questo codice sono rappresentati due cicli:

1. Nel primo ciclo viene variata la temperatura di funzionamento del transistor, successivamente simulato un transitorio e infine salvato il nome del gruppo all'interno del vettore "groupvector"
2. Nel secondo ciclo viene impostato, per ogni iterazione, il gruppo relativo ad ogni simulazione e poi visualizzato sul grafico la corrente del transistor simulato.

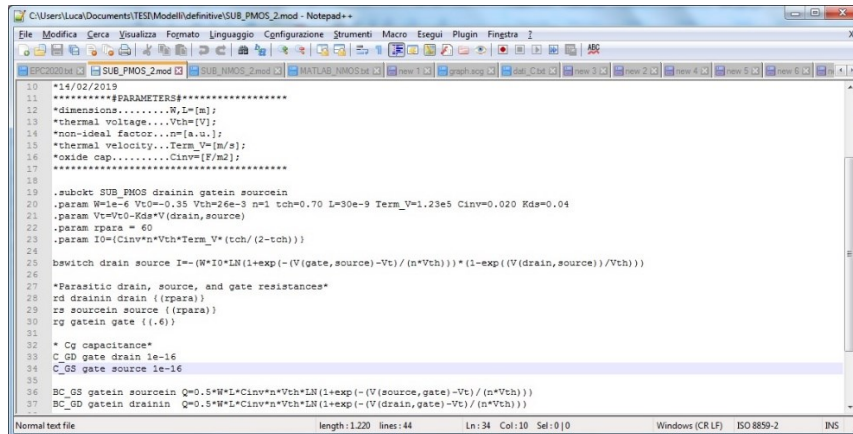
Tale script si riferisce al circuito:



## 2 Modelli e sottocircuiti

### 2.1 Creazione di una libreria

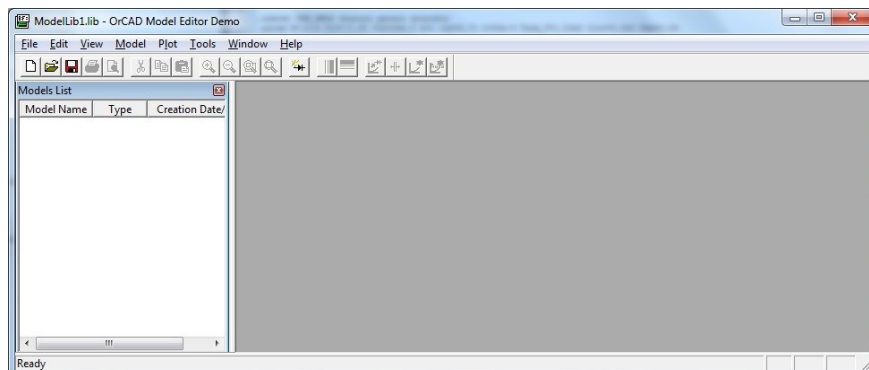
Scrivere il modello del dispositivo con un generico editor di testo come Notepad++ (o simili) e salvarlo nel formato .mod



```
10 *14/02/2019
11 *****PARAMETERS*****
12 *dimensions.....W,L[m]
13 *thermal voltage...Vt=[V]
14 *non-ideal factor...n=[a.u.]
15 *thermal velocity...Term_V=[m/s]
16 *oxide cap.....Cinv=[F/m2]
17 *****
18
19 .subckt SUB_PMOS drain gatein sourcein
20 .param W=1e-6 Vt0=-0.35 Vth0=26e-3 n=1 tch=0.70 L=30e-9 Term_V=1.25e5 Cinv=0.020 Kds=0.04
21 .param Vt=Vt0-Kds*V(drain,source)
22 .param rpara = 60
23 .param I0=(Cinv*n*Vth*Term_V*(tch/(2-tch)))
24
25 bswitch drain source I=-(N*I0*LN(1+exp(-(V(gate,source)-Vt)/(n*Vth)))*(1-exp((V(drain,source))/Vth)))
26
27 *Parasitic drain, source, and gate resistances*
28 rd drainin drain ((rpara))
29 rs sourcein source ((rpara))
30 rg gatein gate ((.6))
31
32 * Cg capacitance*
33 C_GD gate drain 1e-16
34 C_GS gate source 1e-16
35
36 BC_GS gatein sourcein Q=0.5*W*L*Cinv*n*Vth*LN(1+exp(-(V(source,gate)-Vt)/(n*Vth)))
37 BC_GD gatein drainin Q=0.5*W*L*Cinv*n*Vth*LN(1+exp(-(V(drain,gate)-Vt)/(n*Vth)))
```

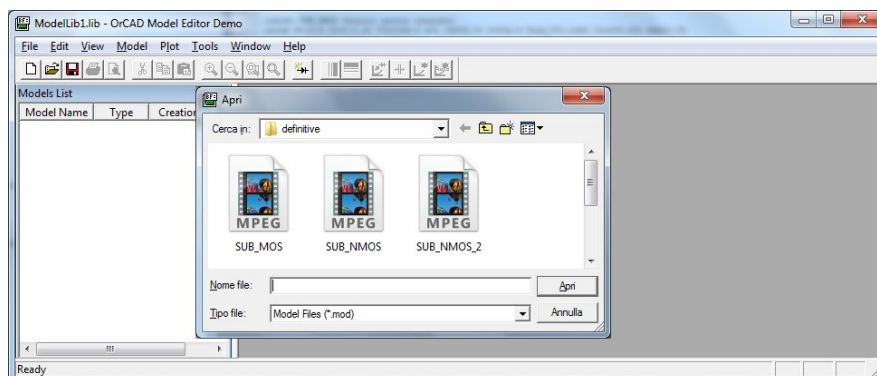
Aprire il programma “Pspice Model Editor Student” o equivalente quindi procedere con creazione di una nuova libreria

**File>New**

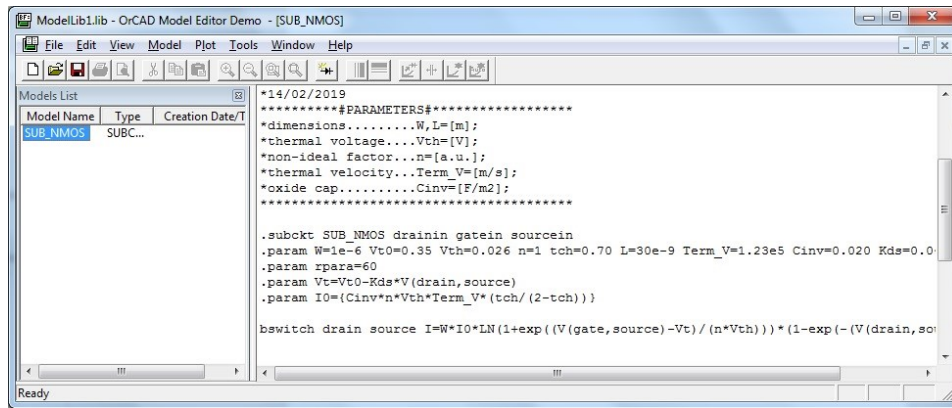


A questo punto basterà importare il file .mod creato con Notepad++:

**Model > Import**



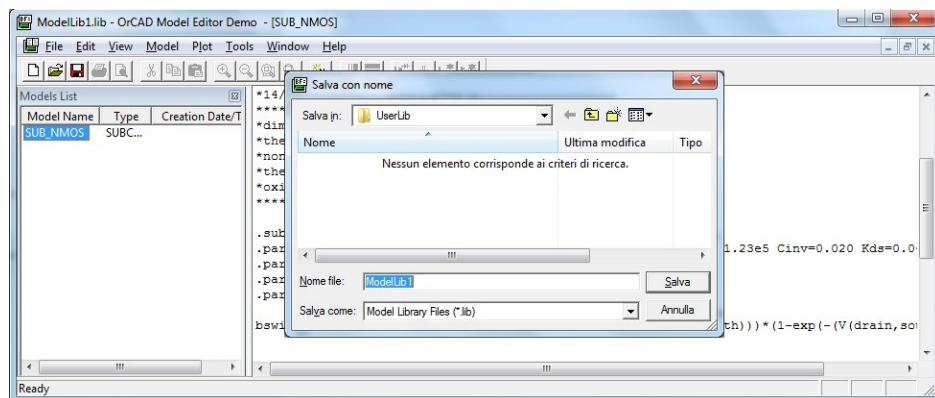
Doppio click sul file .mod di interesse



**NOTA:** Ovviamente se si desidera inserire più modelli all'interno della stessa libreria basterà ripetere i passaggi descritti in questa pagina

A questo punto si può procedere al salvataggio nel formato .lib:

**File > Save As**

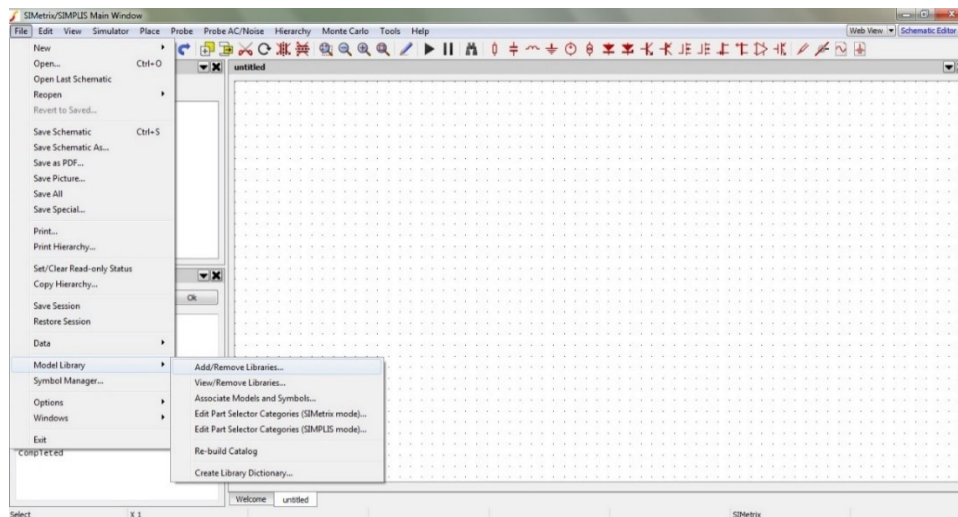


## 2.2 Procedura per il caricamento di una libreria o un modello in SIMETRIX

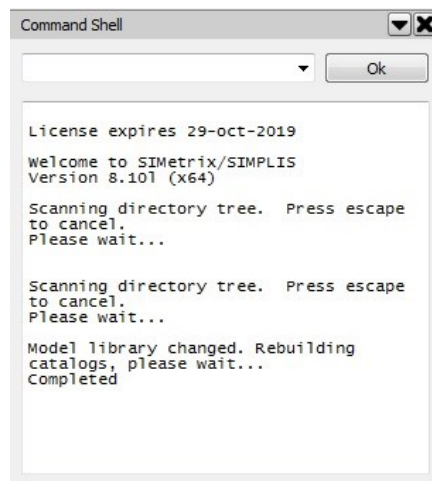
Per inserire un modello o una libreria in Simetrix è necessario andare su:

**File > Model Library > Add/Remove Library**

E selezionare il modello/libreria

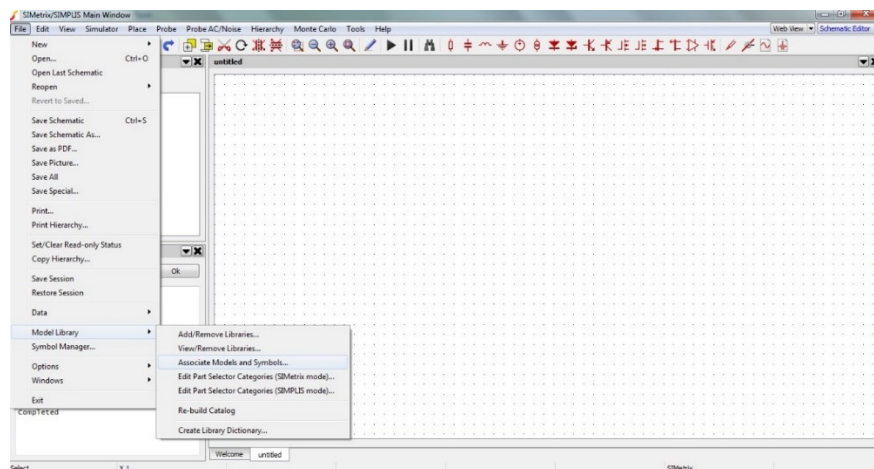


I messaggi visualizzati nella COMMAND SHELL che confermano l'avvenuto caricamento della libreria sono:

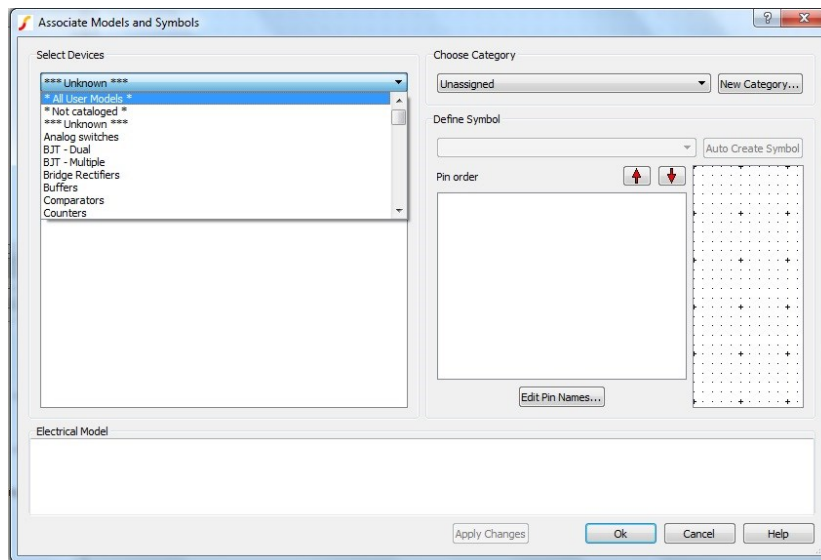


Prima di potere utilizzare il modello, è necessario associare un simbolo:

**File > Model Library > Associate Models and Symbols**



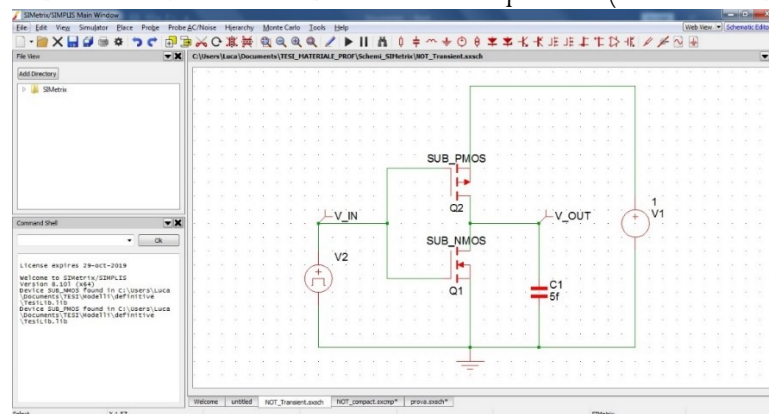
A questo punto apparirà una finestra con l'elenco dei file .mod associati alla libreria appena caricata nel simulatore.



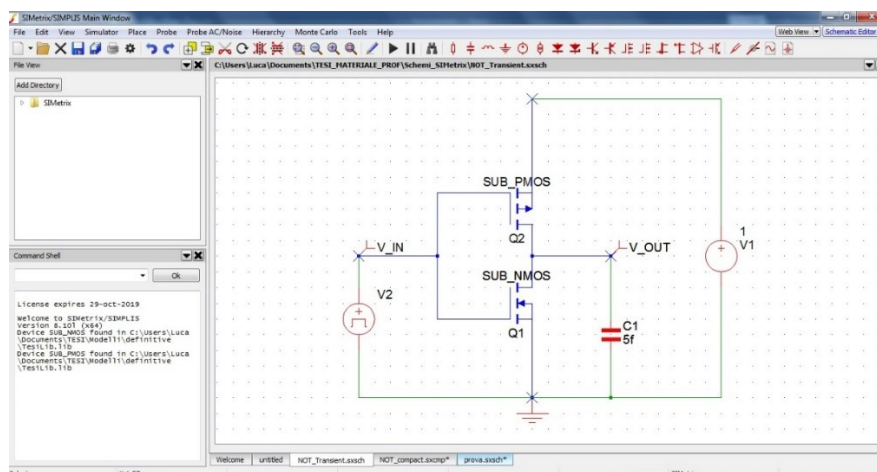
Basterà selezionare il nome del modello che sarà presente alla voce **\*\*\*Unknown\*\*\*** e selezionare il simbolo in **DEFINE SYMBOL**.

## 2.3 Procedura per la creazione di sotto-circuiti

Aprire lo schematico del circuito che si vuole “compattare” (es. circuito invertitore)



Selezionare gli elementi del circuito che si vuole utilizzare per il sotto-circuito e copiarli (CTRL+C)





Aprire un nuovo file:

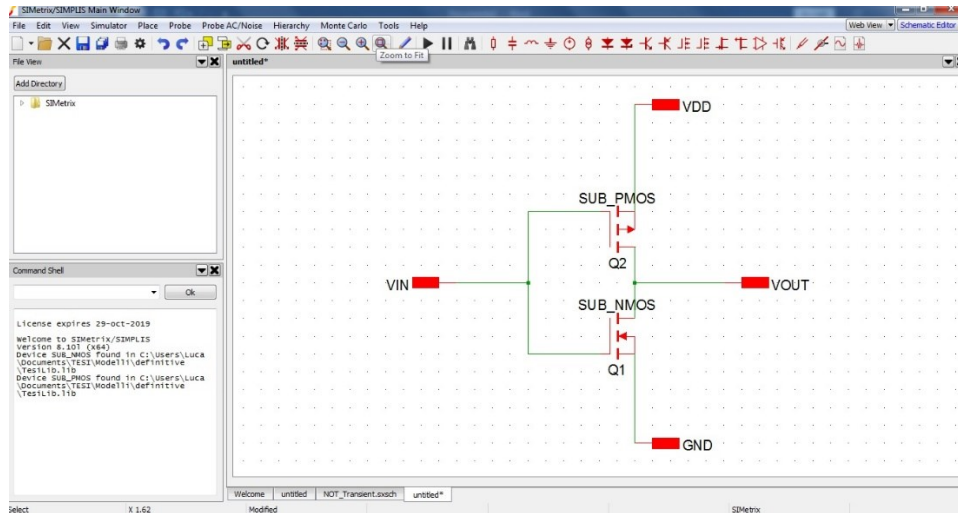
**File > New > SIMetrix Schematic**

Incollare la parte di schematico copiata precedentemente (CTRL+V).

Inserire i pin globali (tasto H):

**Place > Hierarchy > Module Port**

I pin globali permettono di interagire con il sotto-circuito una volta terminato il processo di integrazione e si possono rinominare con un doppio click sul pin stesso.

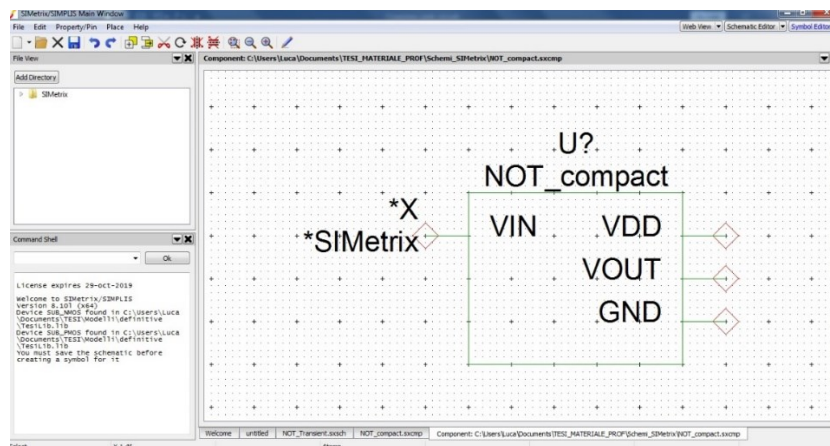


Prima di procedere è necessario salvare il “nuovo” schematico.

Disegnare una forma geometrica del simbolo che sarà visualizzato al posto del circuito nelle future applicazioni:

**Hierarchy > Open/Create Symbol for Schematic (tasto “S”)**

- 1) Non eliminare l’etichetta “U?”, potrebbe dare degli errori di pin flottante durante l’implementazione dei circuiti futuri;
- 2) SIMetrix definisce in modo automatico la disposizione dei terminali globali e la forma geometrica del simbolo come rappresentato nella figura successiva.

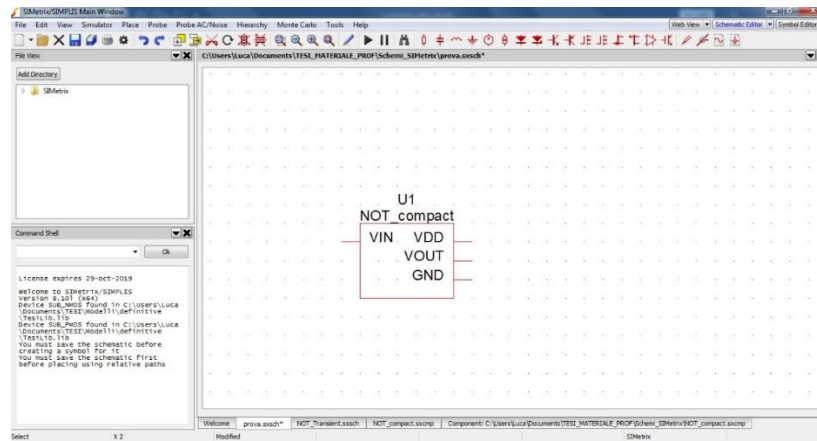


Procedere con il salvataggio del simbolo.

## 2.4 Utilizzo dei circuiti “compatti”

Aprire un nuovo file schematico, salvarlo ed inserire il circuito “compattato” con:

**Place > Hierarchy > Component (Relative Path)**





## 3 Progetto 1: Capacità di gate

### 3.1 Parametri di simulazione

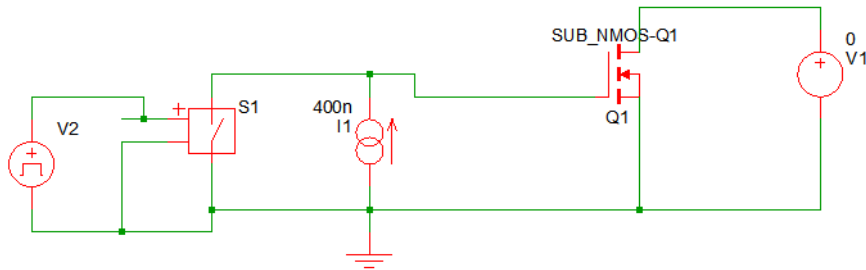
- maxTime = 400p
- NMAX = 50

### 3.2 Parametri del modello

T [K]	W [m]	L [m]	Vt0n [V]	Vt0p [V]	n	tch	Term_V [m/s]	Cinv [F/m <sup>2</sup> ]	Kds	rpara [Ω]
300	1e-6	30e-9	0,35	-0,35	1	0,7	1,23e5	0,02	0,04	60

### 3.3 Circuito e simulazione

Il circuito impiegato per la simulazione della capacità di gate di un transistor fa uso di un generatore ideale di corrente continua che al tempo  $t = 0$  risulta cortocircuitato su un interruttore ideale. Successivamente l'interruttore viene aperto imponendo una corrente costante al terminale di gate dell'NMOS. Tale corrente tende a caricare linearmente nel tempo la capacità di gate, quindi misurando la tensione di gate è possibile ottenere (sapendo che  $I_1$  è costante) la capacità di gate al variare della tensione  $V_{gs}$ .

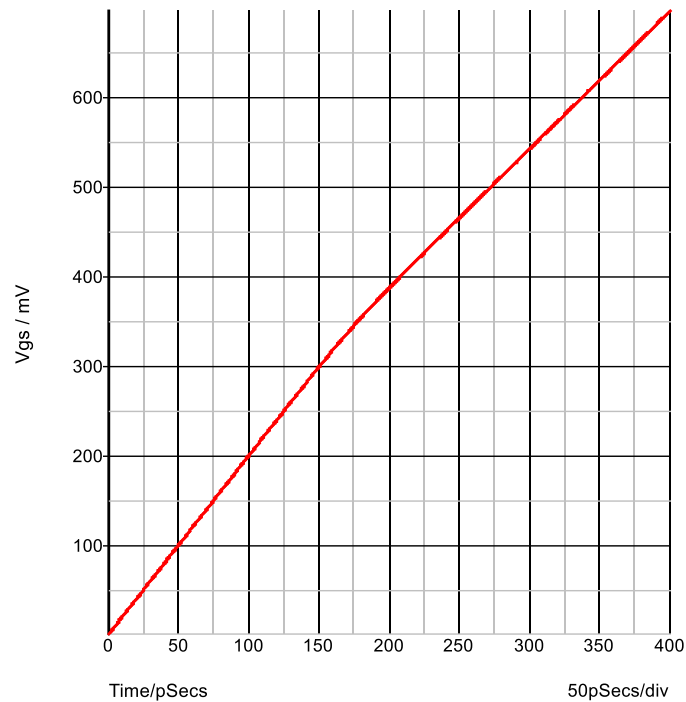


Infatti:

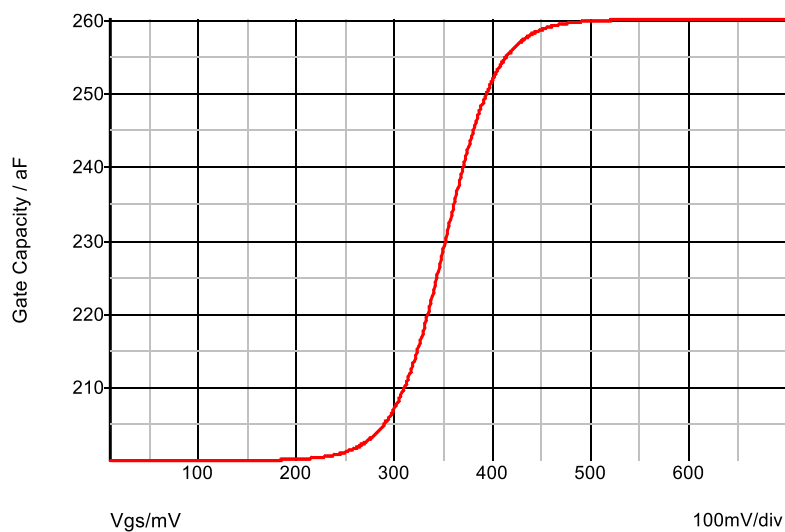
$$C = \frac{\partial Q}{\partial V_G} = \frac{\partial Q}{\partial t} \frac{\partial t}{\partial V_G} = i_G \left( \frac{\partial V_G}{\partial t} \right)^{-1}$$

quindi differenziando la tensione di ingresso, facendone il reciproco e moltiplicando per la corrente di ingresso è possibile ottenere la capacità di ingresso al gate.

La tensione al gate simulata è:



Mentre la capacità simulata ha il comportamento in figura:



Come è possibile notare da questa ultima immagine, a tensioni basse (sottosoglia) la capacità di gate è dominata dai parassiti. Successivamente, quando il transistor va in inversione, la capacità aumenta notevolmente a causa della carica di canale.

## 3.4 Script file

```
ClearMessageWindow
DelGroup /all
close Graph

* Simulation constants
let maxTime = 400p
let NMAX = 50

*Model constants
```

```

let global:T      = 300
let global:W      = 1e-6
let global:L      = 30e-9
let global:n      = 1.3
let global:tch    = 0.70
let global:Term_V = 1.23e5
let global:Cinv   = 0.020
let global:Kds    = 0.2
let global:rpara  = 60
let global:Vt0n   = 0.3
let global:Vt0p   = -0.3

Netlist design.net
Run /noerr /file design.net /an {''.TRAN 0 '& maxTime &' 0 500f'}
Plot Q1.gatein /ylabel "Vgs" /yunit "V" /name "Vgs"

let current = I1#p[10]
let tru = Q1.gatein
let v = diff(tru)
let bias = Truncate(v, 5p, maxTime)
let cg = Truncate((bias^-1 * current), 5p, maxTime)
setRef cg Truncate( Q1.gatein, 5p, maxTime)
Plot cg /name "Gate Capacity" /Yunit "F" /xlabel "Vgs"

```

## 4 Progetto 2: Caratteristica I/O di invertitore

### 4.1 Parametri di simulazione

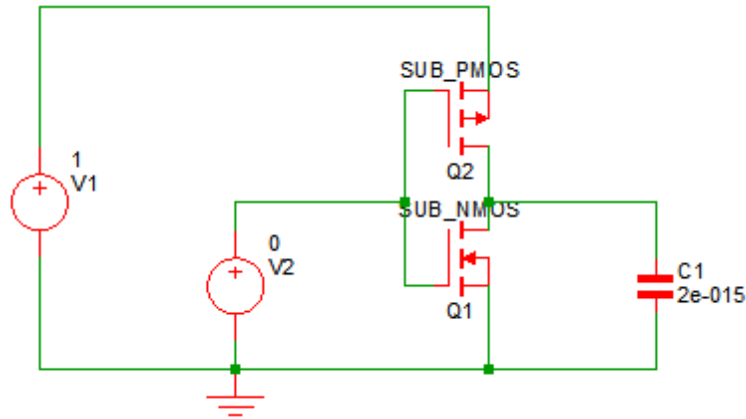
- VDD = 1
- stdCap = 2f
- minVGSstep = 10m
- maxVDD = 1
- minVDD = 0.1
- NVDD = 10
- maxKDS = 0.2
- NKDS = 5

### 4.2 Parametri del modello

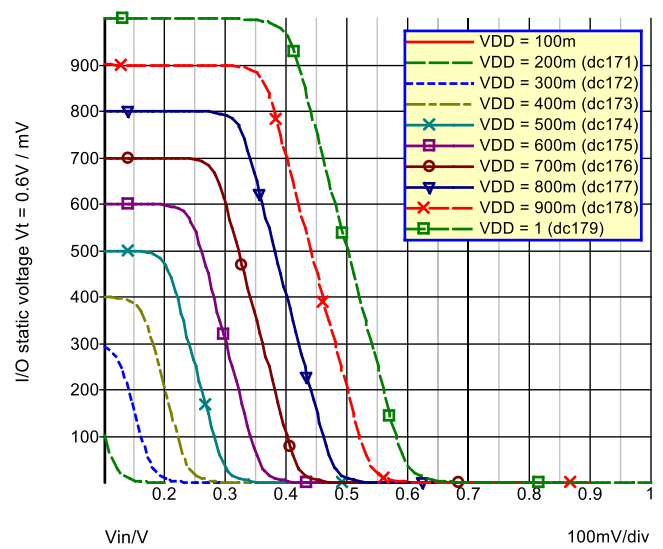
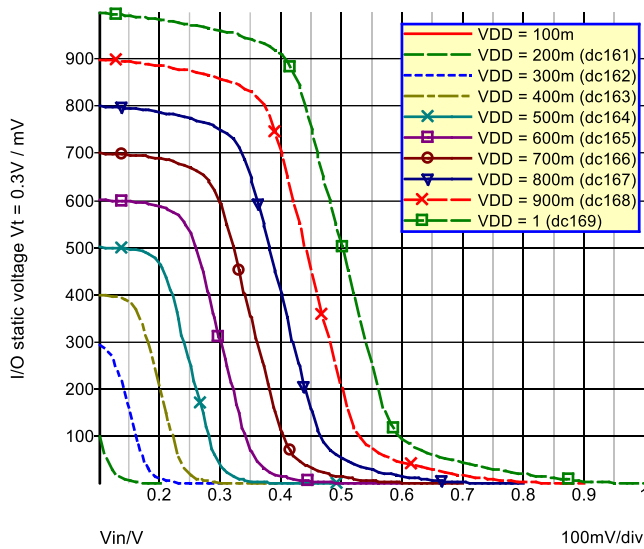
T [K]	W [m]	L [m]	Vt0n [V]	Vt0p [V]	n	tch	Term_V [m/s]	Cinv [F/m <sup>2</sup> ]	Kds	rpara [ $\Omega$ ]
300	1e-6	30e-9	0,3	-0,3	1.3	0,7	1,23e5	0,02	0,2	60

### 4.3 Circuito e simulazione

Per la simulazione della porta NOT, e la comprensione di come variano le caratteristiche di tale porta logica in relazione alla variazione dei parametri dei transistori, è necessario utilizzare il circuito mostrato nella figura precedente:



L'analisi più semplice da operare sulla porta not è quella della transcaratteristica ingresso-uscita. In particolare, si può notare dai seguenti grafici come tale transcaratteristica abbia profili più netti al crescere della  $V_{t0}$ .



## 4.4 Script file

```
ClearMessageWindow
DelGroup /all
close Graph
* VDD voltage
let global:VDD = 1
* Standard output capacity
let global:stdCap = 2f
* Max step simulation
let global:minVGStep = 10m
* MAX VDD to analyze
let maxVDD = 1
* MAX VDD to analyze
let minVDD = 0.1
* Number of VDD to analyze
let NVDD = 10
* VDD vector, defines also the number of VDD
let VDDs = vector(NVDD)
* DIBL PARAMETERS
* MAX VDD to analyze
let maxKDS = 0.2
* maximum resistance of transistor
let maxR = 350
* Number of KDS to analyze
let NKDS = 5
* VDD vector, defines also the number of VDD
let KDSs = vector(NKDS)
```

```

* Number of resistance to analyze
let NR = 5
* VDD vector, defines also the number of VDD
let Rs = vector(NR)
*Propagation time of each VDD, must be bigger than VDD, elements in the ranges of VDDs
*must be refreshed the others remain 0
let propTime = vector(NVDD)
* Vector to store data
let groupvector = ['', '', '', '', '', '', '', '', '', '']

Unselect
Select /prop Ref C1
Prop value {stdCap}
** GLOABL VALUES FOR TRANSISTORS

let global:T = 300
let global:W = 1e-6
let global:L = 30e-9
let global:n = 1.3
let global:tch = 0.70
let global:Term_V = 1.23e5
let global:Cinv = 0.020
let global:Kds = 0.2
let global:rpara = 60
let global:Vt0n = 0.3
let global:Vt0p = -0.3

**STATIC ANALISYS
for i = 0 to 1
  if i== 1 then
    let global:Vt0n = 0.6
    let global:Vt0p = -0.6
  endif

  for newVDD = 0 to NVDD-1
    let VDDs[newVDD] = maxVDD / NVDD * (newVDD + 1)
    let global:VDD = VDDs[newVDD]
    Unselect
    Select /prop Ref V1
    Prop value {VDD}
    Netlist design.net
    Run /noerr /file design.net /an {'DC V2 '&str(minVDD)&' '&str(VDD) &' ' &str(minVGSstep)}
    let groupvector[newVDD] = (Groups())[0]
    KeepGroup {groupvector[newVDD]}
  next newVDD
** SHOW RESULTS
for newVDD = 0 to NVDD-1
  SetGroup {groupvector[newVDD]}
  if newVDD == 0 then
    Plot Q2.drainin /name {'VDD = ' & VDDs[newVDD]} /ylabel "I/O static voltage Vt = 0.3V" /xlabel "Vin"
  else
    Curve Q2.drainin /name {'VDD = ' & VDDs[newVDD]}
  endif
next newVDD
next i
exit all

```