# DEPFET simulation model

A. Wassatsch and R. H. Richter

- Motivation
- Why choose CMI ?
- The model implementation
- Testcase
- How to use the model

mpi halbleiterlabor

Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)

# Motivation

For SuperBelle and ILC we are promising a very fast operation of very huge sensor arrays.

But this means:

5cm long readout lines, huge capacitive loads on all lines (control and readout), non 0 Ohm lines

We should ask ourself whether this sets fundamental speed limits and which are the screws to turn (by design and by technology) to optimize the system.

We want to provide a DEPFET and sensor array model to the ASIC designers, to be able to simulate the whole chain:

Switcher → Sensor → ADC

# CMI versus Verilog-A

Compiled Model Interface

Pro

- Plain C code
- External standard libs usable (gsl)
- High achievable simulation speed (only convertion function for the io with the spectre simulation kernel)
- Small memory foot print for variables

Contra

- Routines are executed in the same spectre simulation kernel thread (no benefitz on multiprocessor plattforms)
- Interface is spectre simulator specific

Verilog-A

Pro

- Use of an additional simulation kernel thread (speedup on multiprocessor plattforms)
- Code more or less portable to other simulators as spectre

Contra

- Limited interpolation routine support
- Slowdown due to the additional translation overhead (verilog-A → C → exec code)
- Large overhead for memory fields

→ we decide to use CMI

# The depfet CMI model implementation

- Derived from the mosx CMI example provided by cadence
  - elimination of any noise related program code (not needed at the moment)
  - At the end approx 30 pages of program code, but most of them for the interface to the spectre simulator which can be easily adapted to our needs
  - Only two relevant functions hlldepfetTranEvalResidualOnly and hlldepfetTranEval has to be adapted to describe the special depfet model behavior for the simulator
  - An additional input data filter for the depfet measurement table file was added
- Currently the model is limited to a single readout sequence (measure,clear,measure)
- Model object orientated implementation: each instantiated depfet model in a spectre circuit can have his one measurement table and it's own state

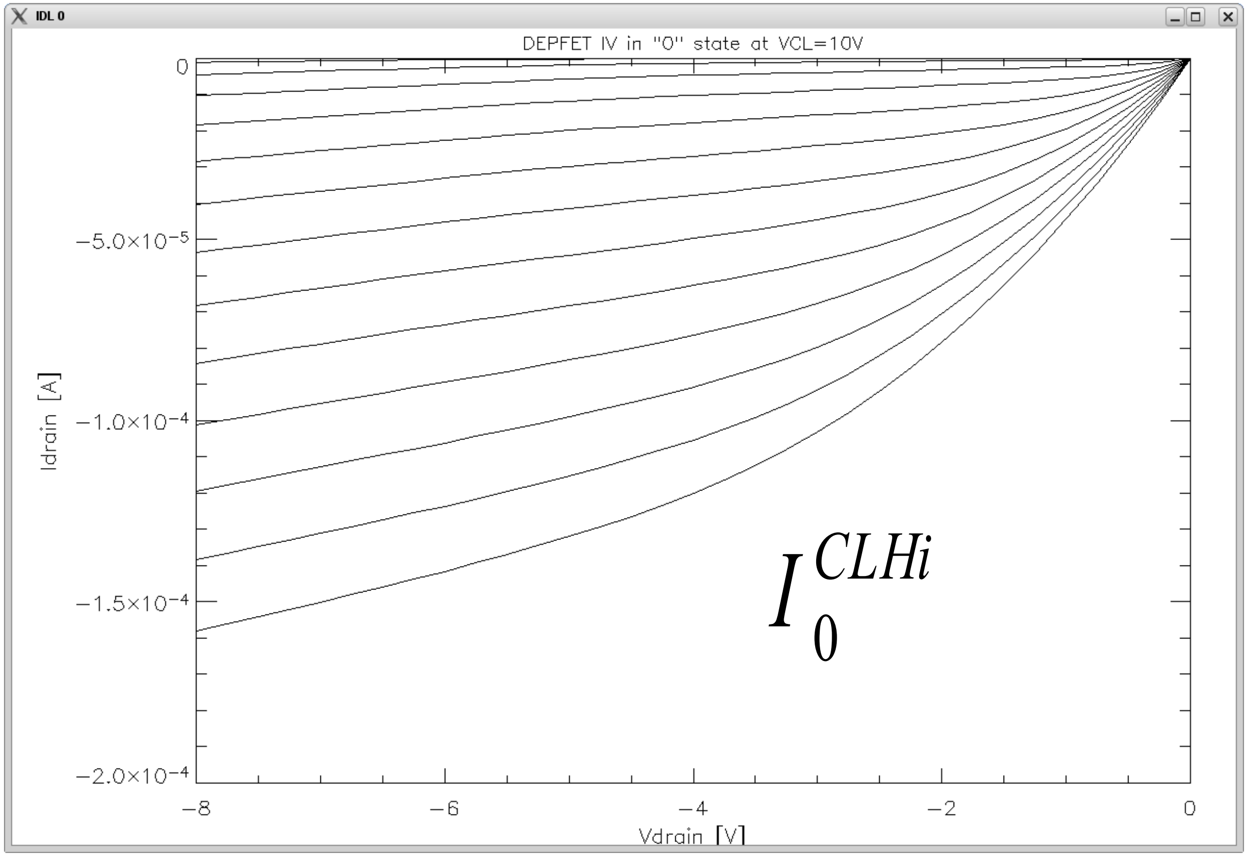
# Details of the model implementation

- 2 dimensional lookup table for the cleared $I_{DS} = F(V_G,V_{DS})$
- Spline interpolation for the values which are not in the table via gsl function
- Correction of the transconductance of the clear gate on $I_{DS}$ $(g_m)$
- Implementation of the charge induced effect on $I_{DS}$ $(g_q)$
- Modeling of the clear behavior

for questions or a more detailed description please ask

Rainer !

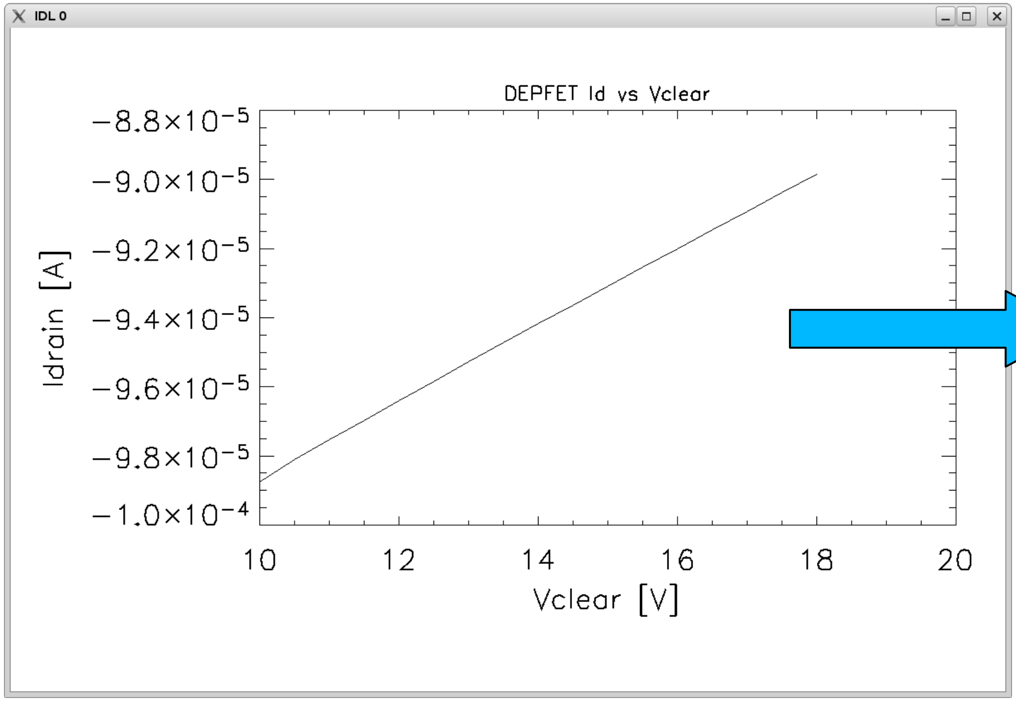
# Lookup table model




Measured at $V_{cl\ high}$ = 10V (to keep internal Gate empty during static measurements)

For read mode $V_{cl\ low}$: correction of currents for Clear voltage needed

# Transconductance of the Clear electrode on drain current




$@V_G = V_D = -5V$

$g_m = 1.1\ \mu S$

New lookup table for $V_{cl}$ :

$$I_0^{CLLo} = I_0^{CLHi} + \sqrt{\frac{I_0^{CLHi}}{I_{0Ref}^{CLHi}}}\, g_m^{CL}\left(10\,V - V^{CLLo}\right)$$

$$I_{0Ref}^{CLHi} = I_0^{CLHi} @ V_G = V_D = -5\ V$$

## 'Filling' the Internal Gate

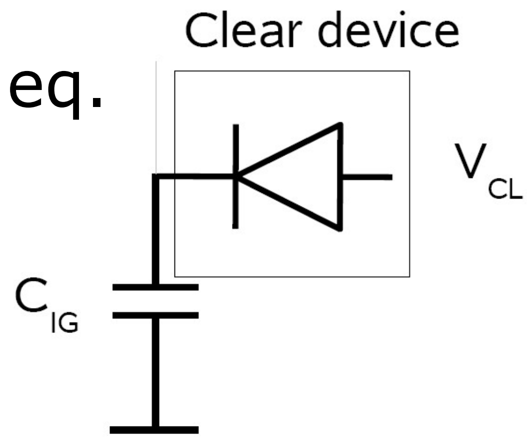
$$I_1^{CLLo} = I_0^{CLLo} - \sqrt{\frac{I_0^{CLHi}}{I_{0Ref}^{CLHi}}}\; g_q \; N_{SIG}$$

$g_q$ internal amplification @$V_G$=$V_D$=-5V

400pA/electron

# Clear implementation

$$\frac{\Delta Q}{\Delta t} = I_{C0}\left(e^{y} - 1\right)$$ Ebers-Moll like diode eq.




$$y = k\left(V_{CL} - V_{ON} - V_{IG}\right) / V_T * \frac{V_{IG}^{0} - V_{IG}}{V_{IG}^{0}}$$

$$V_{IG} = \frac{Q + \Delta Q}{C_{IG}}$$ Internal Gate Potential

$V_{IG}^{0}$ empty Internal Gate Potential

$I_{C0}$ saturation current

$k$ constant

$V_{ON}$ onset voltage for the clear process (potential barrier)

$V_T$ temperature voltage

stops clear current when the Internal Gate is empty

However, we have a transcendental function and need some numerics!

# Clear implementation (cont.)

Spectre provides a time step $\Delta t$
The model has to calculate the corresponding $\Delta Q$ at the applied $V_{CL}$
The new signal charge change $I_d$ which is returned to the network.

$$\frac{\Delta Q}{\Delta t} = I_{C0}\left(e^{y} - 1\right)$$

$$f = I_{C0}\left(e^{y} - 1\right) - \frac{\Delta Q}{\Delta t}$$

Search for the root (null) gives $\Delta Q$

# Clear implementation (cont.)

Newton-Raphson iteration

$$\Delta Q_{new} = \Delta Q_{old} - \frac{f}{f'}$$

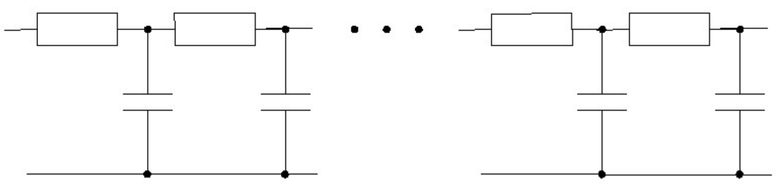
Very efficient!

Usually 4 iterations to reach a precision of 10As

It‘s easy to implement more refined Clear models
 requirements for f : derivative, monotony

# Superbell depfet testcase Example

Each line is a distributed RC line of 50 segments



worst case pixel

Control lines

Switcher: $R_{down}$ 30 Ohm, $R_{up}$ 52.5 Ohm
Gate line: $R_{total}$ = 40 Ohm, $C_{total}$ = 50pF
Clear line: $R_{total}$ = 40 Ohm, $C_{total}$ = 50pF

Readout (drain) line

$R_{total}$ = 215 Ohm, $C_{total}$ = 50pF

- single depfet simulation
- matrix edge devices (worts case)
- sequential readout optimisation (minimization of the readout time by overlap of the single sequenzes)

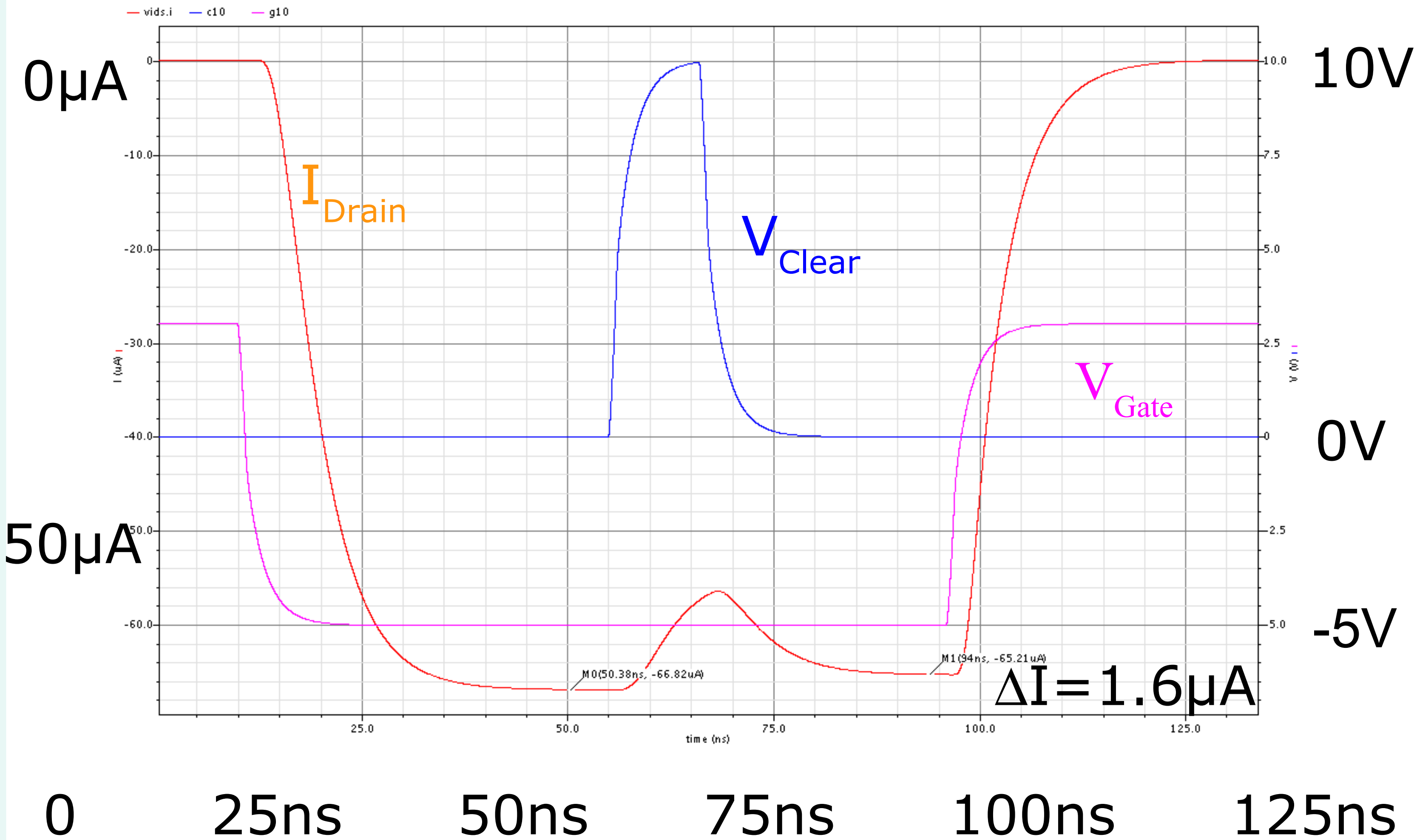
# Spectre example

```
1.    …
2.    * Clear line
3.    rgl ( vc vcr gnd ) subRcline cnt=5 rsum=40 csum=50p
4.    * Drain line
5.    rdl ( vd vdr gnd ) subRcline cnt=5 rsum=170 csum=50p
6.    ...
7.    *  D G  S Cl (Depfet) , look up table in Fpath
8.    mdepfet (vdr vg vs vcr) hlldepfetDevice m=1 InstId=1 Fpath="test.txt" Ns=4000
      Gq=-400e-12 Vcl=0
9.    model hlldepfetDevice hlldepfet
10.   …
```

```
source ........../mmsim6.2/start.sh
cd ...../src
make -f Makefile-64
spectre -cmiconfig cmiConfig -h
spectre -cmiconfig cmiConfig -h hlldepfet
spectre +spice +debug +mt -cmiconfig cmiConfig -format sst2
../test/hlldepfet2.ckt
```

# Read-Clear-Read Cycle

$N_{SIG} = 4000e$



# Tested environment

Spectre from

- mmsim6.2
- mmsim7.01hf076 (actual europractice version 2008/09)

On opensuse 11.0 and centos 4.7 (x86_64bit)

GNU scientific library version 1.9-144.1

# Next steps

- Implementation of a charge injection model for multiple readout sequences
  - Additional terminal, translate input voltage to a corresponding charge deposited in the depfet if clear is not active
  - Charge value will not be integrated, direct control of the actual charge value
- Evaluation of the model on the xfel input stage (with Gulio deVita)
- Refinement of the Superbell test case
  - Switcher output stage model ?
  - DCD input stage model ?

- Comments and Suggestions ?