A Software Package for Numerical Simulation of Semiconductor Devices under HPM Environment

Author: Gong Ding

Supervisor: Wang Jianguo

- Background
- Theory of Semiconductor Physics
- The Numerical Methods in Semiconductor Device Simulation
- GSS, A General-purpose Semiconductor Simulator
- Further Works

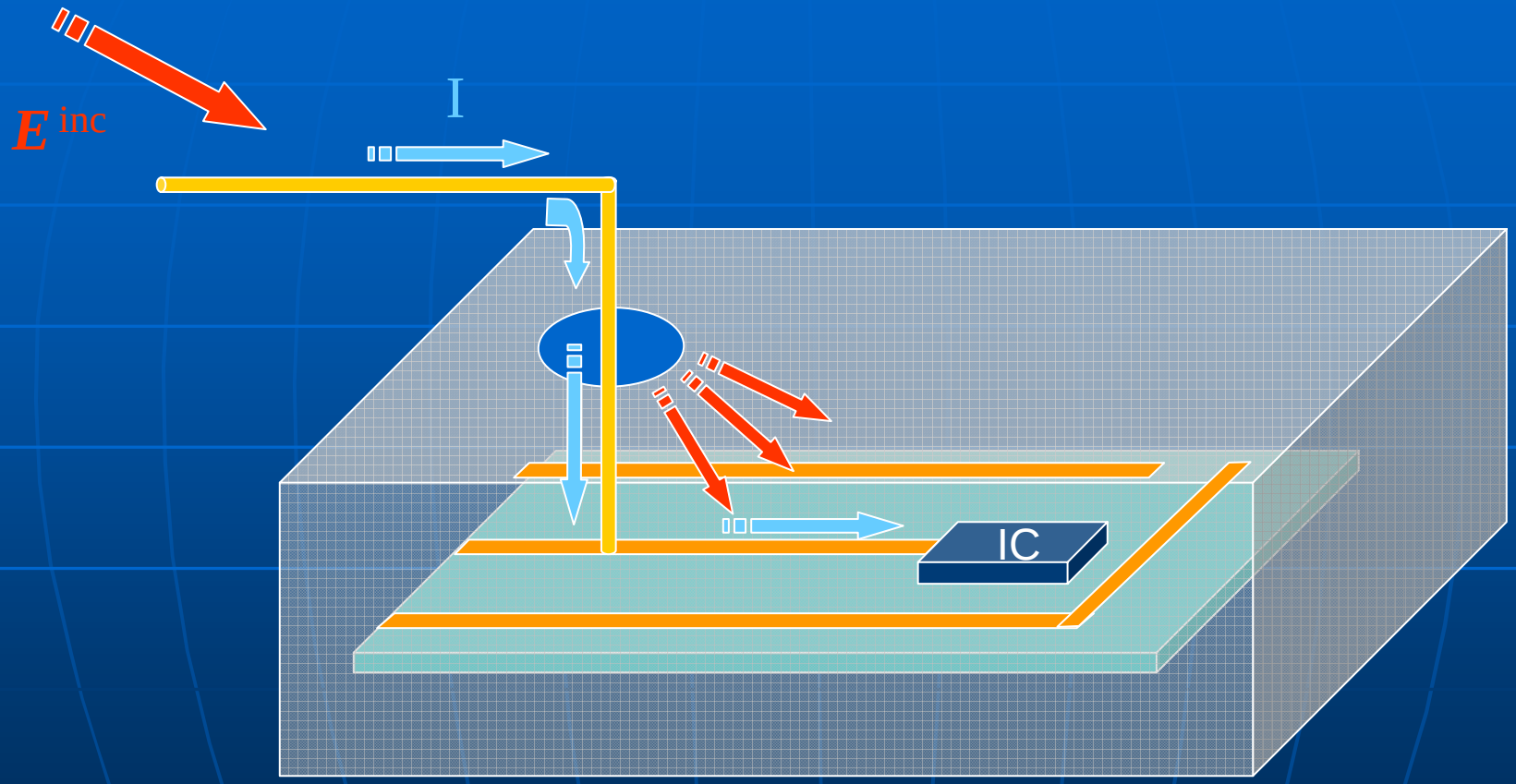
Part1 Background

HPM Weapons

E-BOMB(US)

RANETS-E (Russia)

Part1 Background

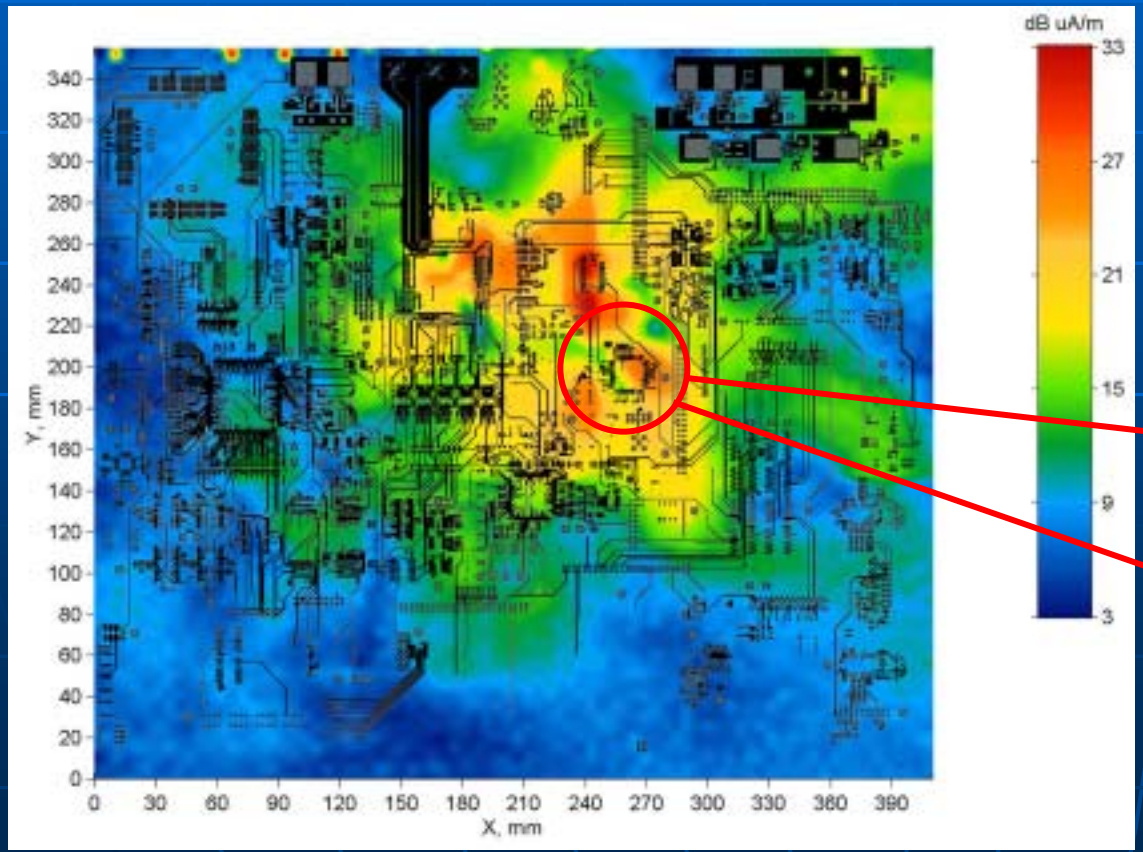


Various ways of incident HPM wave coupling to an electronic system

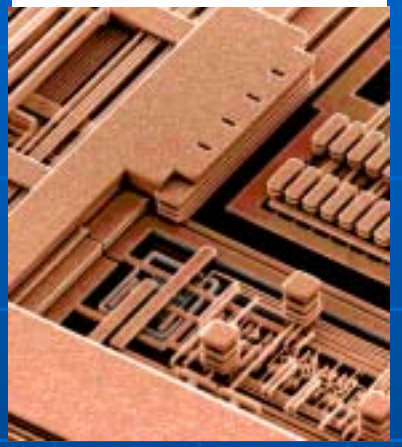
Part1 Background

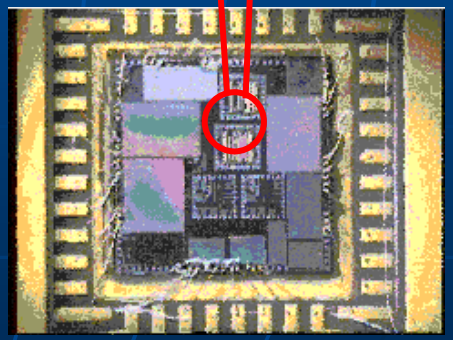
HPM energy flows into transistors

Printed Circuit Board



Transistor





Chip

Part1 Background

Because the characteristic length of semiconductor device is relatively smaller than the wavelength of HPM, the energy deposit can be neglected.

Unfortunately, PCB wires are efficient antennas which can receive HPM energy.

In this thesis, transistors only stimulated by current or voltage sources come from chip pins.

Part2 Semiconductor Physics

S. M. Sze, *Physics of Semiconductor Devices*, 2nd ed. (New York: John Wiley & Sons, 1981).

- Band structure
- Fermi-Dirac distribution
- Classical Boltzmann Transport Equation

Part3 Numerical Methods

- Governing Equations
- Finite Volume Method
- Discretization
- Boundary Conditions
- Nonlinear Solvers
- Linear Solvers

Governing equations of HDM

We only present equations for electrons here

$$\frac{\partial n}{\partial t} + \nabla \cdot (n\vec{v}) = -R$$

$$\frac{\partial (m^* n\vec{v})}{\partial t} + \nabla \cdot (m^* n\vec{v}\vec{v}) + \nabla (nkT) = -en\vec{E} - \frac{m^* n\vec{v}}{\tau_p}$$

$$\frac{\partial (nw)}{\partial t} + \nabla \cdot (n\vec{v}w) + \nabla \cdot (nkT\vec{v}) = -en\vec{v} \cdot \vec{E} - \frac{nw - 2/3nkT_L}{\tau_w}$$

$$\nabla^2 \varphi = -\frac{\rho}{\varepsilon}$$

If we simplify HDM, just drop energy equation and the first two items of moment equation,that is DDM.

Conservation form

The governing equations can be present in conservation form.

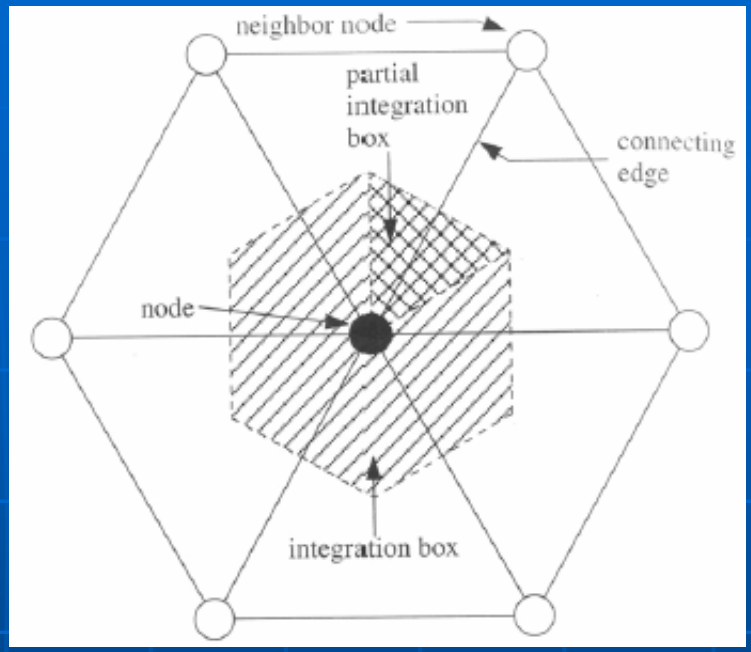
$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{F}_i}{\partial x_i} = \mathbf{S} \cdot \mathbf{Q}$$

$$\mathbf{Q} = \begin{pmatrix} n \\ m^* n\vec{\mathbf{v}} \\ nw \\ 0 \end{pmatrix} \qquad \mathbf{F} = \begin{pmatrix} n\vec{\mathbf{v}} \\ m^* n\vec{\mathbf{v}}\vec{\mathbf{v}} + nkT \\ nw\vec{\mathbf{v}} + nkT\vec{\mathbf{v}} \\ \nabla\varphi \end{pmatrix}$$

$\mathbf{S} \cdot \mathbf{Q}$ is the source item

Finite Volume Method

Use Gauss's law, the integration of the partial item over the control volume can be replaced by boundary integration. That is



$$\frac{\partial}{\partial t}\int \mathbf{Q}dV + \oint \mathbf{F}_i \cdot d\mathbf{s} = \int \mathbf{S}\cdot\mathbf{Q}dV$$

Or the semi-discrete form:

$$\frac{d\bar{\mathbf{Q}}}{dt}V_{cell} + \sum \mathbf{F}_i \cdot l = \bar{\mathbf{S}}\cdot\bar{\mathbf{Q}}V_{cell}$$

$\bar{\mathbf{Q}}$ $\bar{\mathbf{S}}$ is the average value of the cell.

Discretization

How to get the flux at the boundary of cell is the key problem of FVM. A good HD scheme must satisfy high resolution, non-oscillation and total variation diminishing.

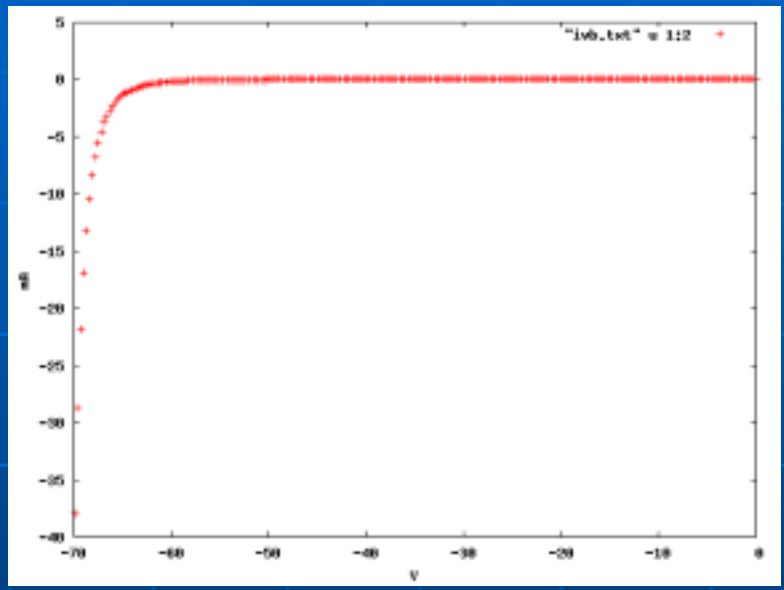
In modern CFD world, least-squares reconstruction with limiters, flux upwind-split and dual-time implicit stepping methods are widely used. Each of them has an uncountable number of papers.

Boundary Conditions

- Ohmic BC
- Schottky BC
- Gate Contact of MOS Structure
- *Current Boundary Condition

Current Boundary Conditions

The Newton's method fails to get convergence at the breakdown region if voltage boundary condition is used.



Therefore, current BC must be applied to satisfy the requirement of HPM simulation.

Nonlinear Solvers

The key process of semiconductor simulation is how to solve the large scale, nonlinear equations arising from discretization step. A flexible, stable and fast arithmetic must be implemented.

- Basic Newton
- Line Search
- Trust Region

Newton's Method

$$f(x) = 0$$

$$\mathbf{J}_k \delta_k = -f(x_k)$$

$$x_{k+1} = x_k + \delta_k$$

Fast arithmetic: quadratic convergence

But only convergence when $x \to x^*$

Only used in IV curve tracing or transient simulation.

Line Search

$$\mathbf{J}_k \delta_k = -f(x_k)$$

$$x_{k+1} = x_k + \alpha \delta_k$$

The first equation only determine the descent direction. Then the step size alpha is solved by a safeguarded polynomial interpolation of f(x).

This arithmetic only holds when Jacobian matrix is exact or nearly exact. GSS tried its best to get exact Jacobian matrix, so line search is the default nonlinear solver for get a initial solution.

Trust Region

$$\mathbf{B}_k = \mathbf{J}_k + \lambda \mathbf{I}$$

This method uses positive definite matrix B instead of J to reduce the step size. A sufficient small step size may satisfy toyler series and get convergent gradually.

Early edition of GSS didn't have exact Jacobian matrix, trust region was applicable. But now, line search method is recommended.

Linear Solvers

All the Nonlinear solvers request a fast linear solver.

In most of the situation, an approximate linear solver is enough.

- LU Factorization Method
- Fixed Iterative Method (GS,SOR,SSOR)
- Krylov Subspace Method (CG,GMRES,BICG)

Part4 Introductions of GSS

- Pre-processor
- Model File Format
- Command File Syntax
- Flexible Material Database
- Build-in Solvers
- Post-processor

GSS 0.42

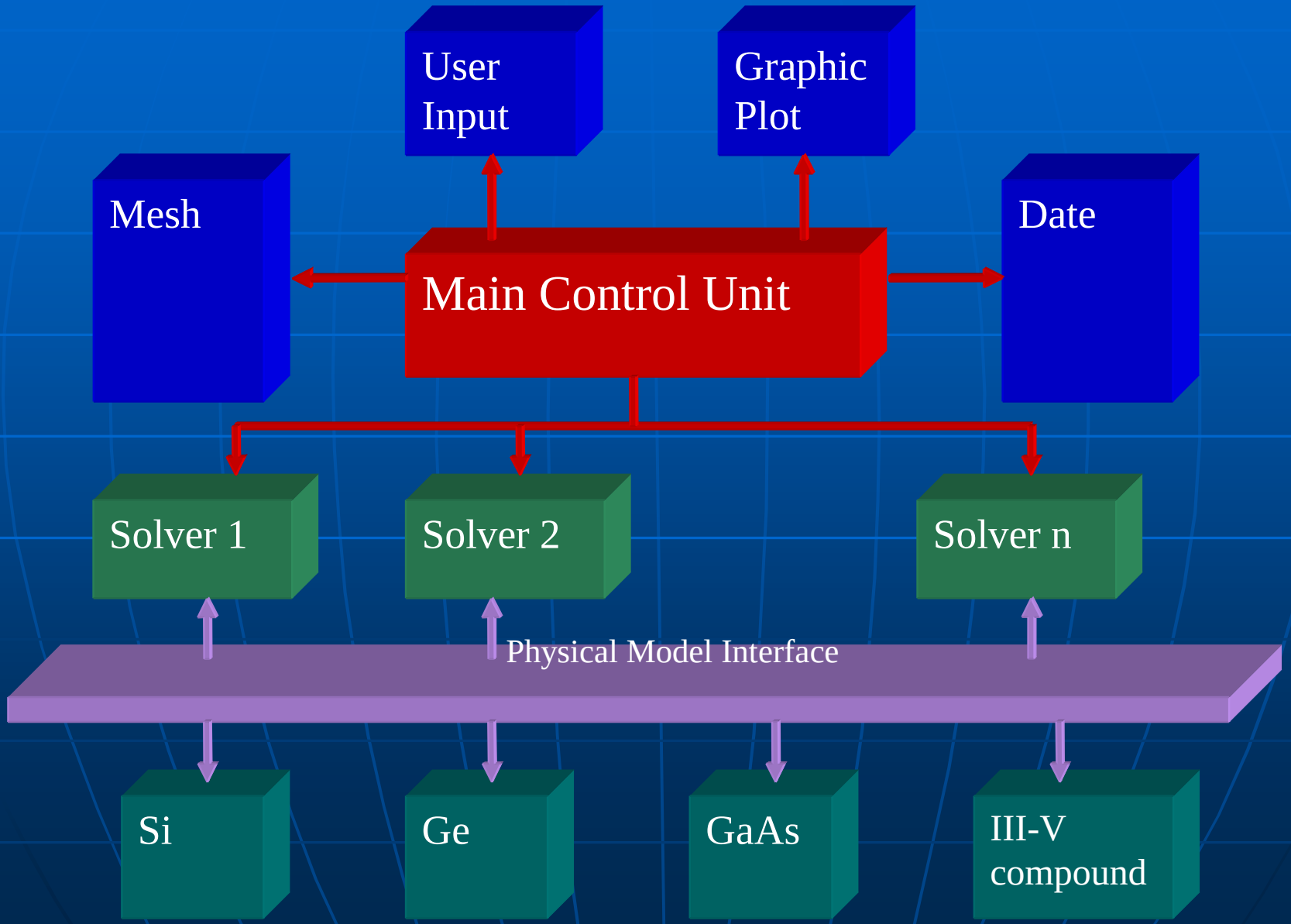
- What is GSS?

  GSS is a General-purpose Semiconductor Simulator. Besides several libraries, version 0.42 has more than 20000 lines of c++ code.

GSS 0.42 have these features

- Use CGNS as standard input/output file format
- Run time parameters are specified by cmd file
- Unstructured mesh(triangle/rectangle) support
- Adaptive mesh refinement
- Two build-in solvers : DDM and HDM

Software Structure of GSS

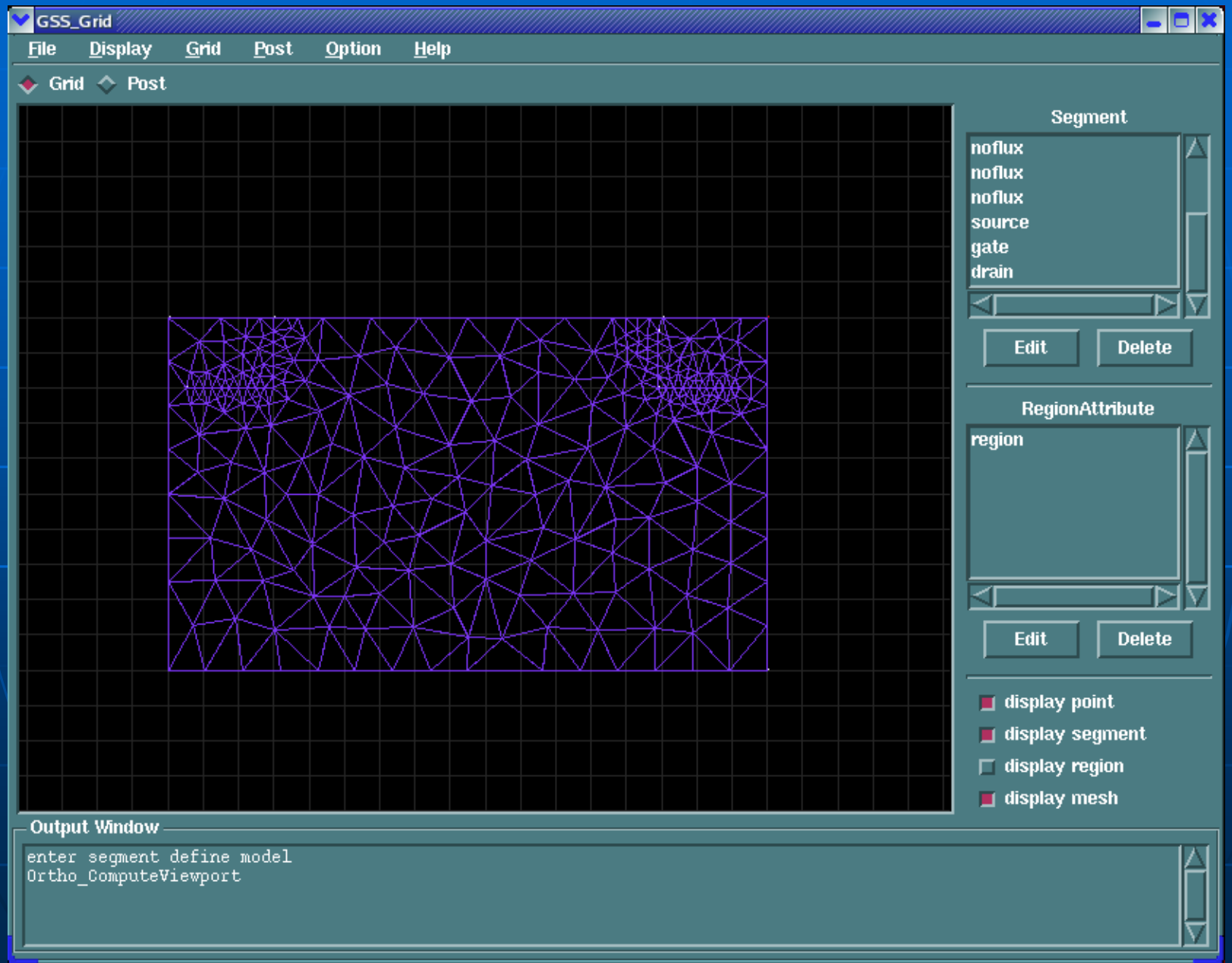


Pre-Processor

GSS support Medici compatible model description language, which can build device model, do mesh division and adaptive mesh refinement easily.

Beside that, some auxiliary tools can be used as pre-processor.

SGrid: Another Pre/Post Processor



Interface to Other Software

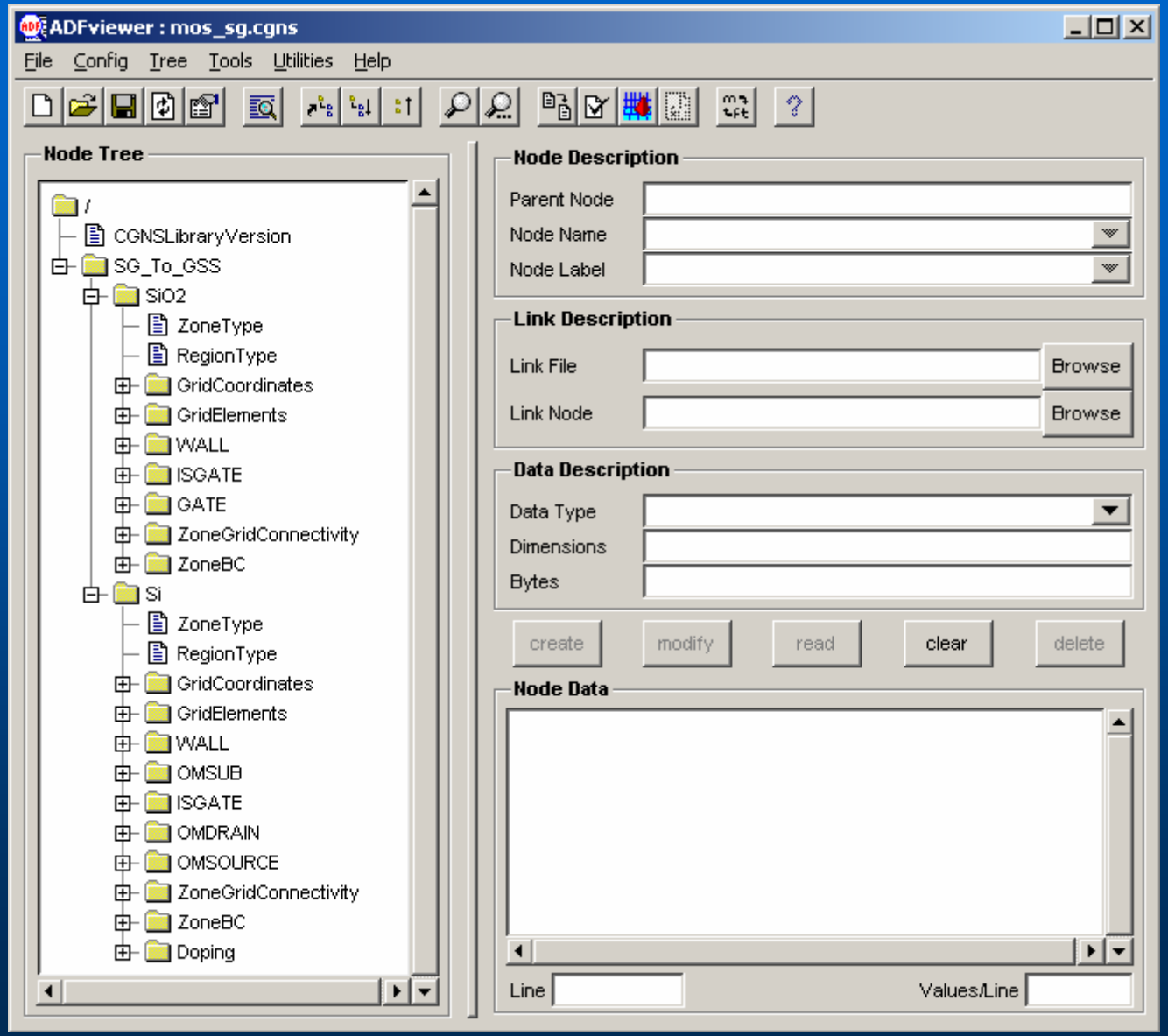
GSS can employ Sgframework or Medici to generate device description file. While a small tool TIFTOOL can convert Medici TIF file to CGNS file, which can be read by GSS.

CGNS, The I/O File Format

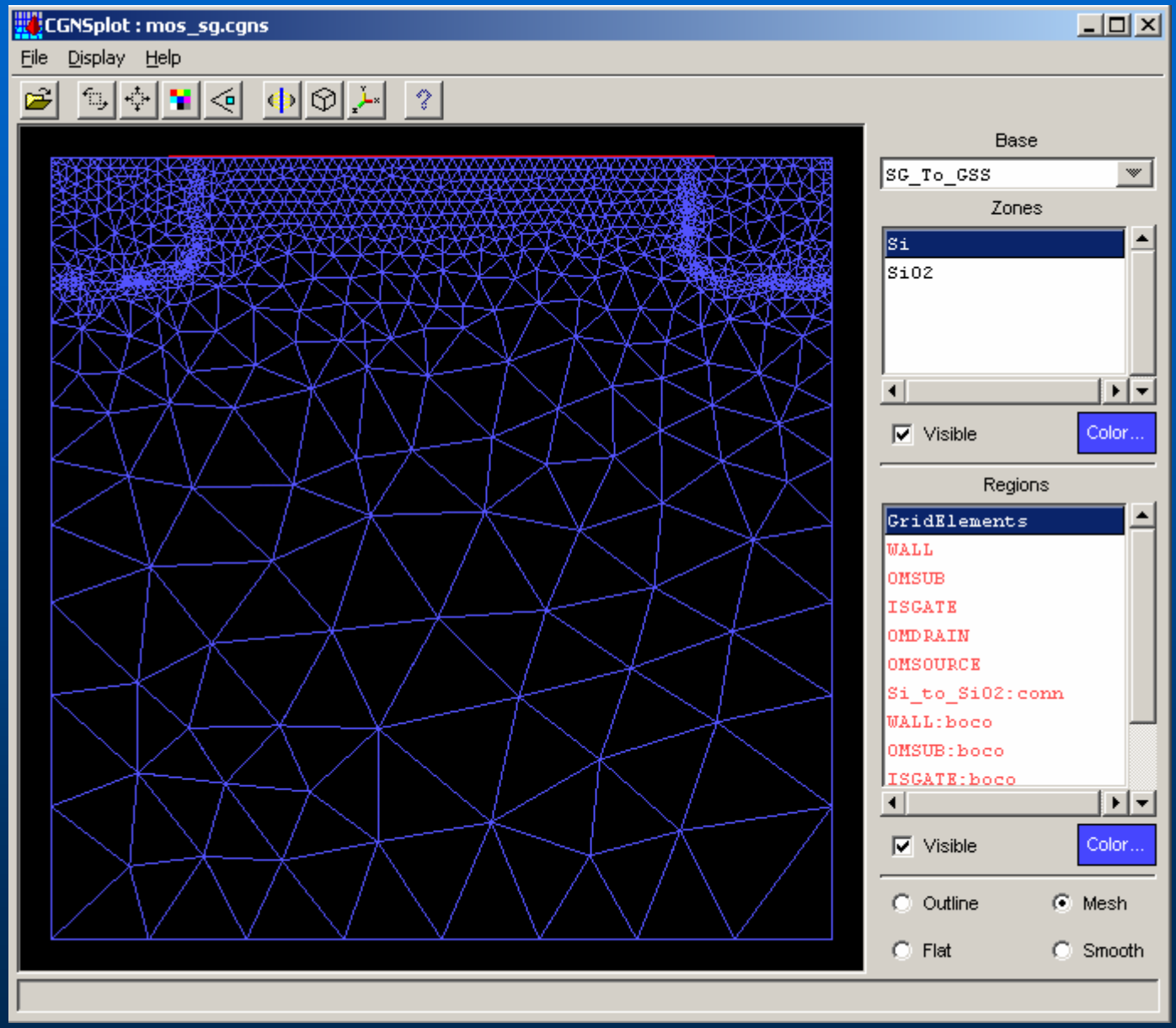
CGNS: CFD General Notation System

- Supported by NASA and many commotional CFD Corporations.
- Mesh, boundary condition and solution data are stored in one file.
- Freeware Adfviewer and Cgnsplot can help for debugging.
- CGNS is well supported by ICEM CFD10.0, the world’s top pre/post processor.

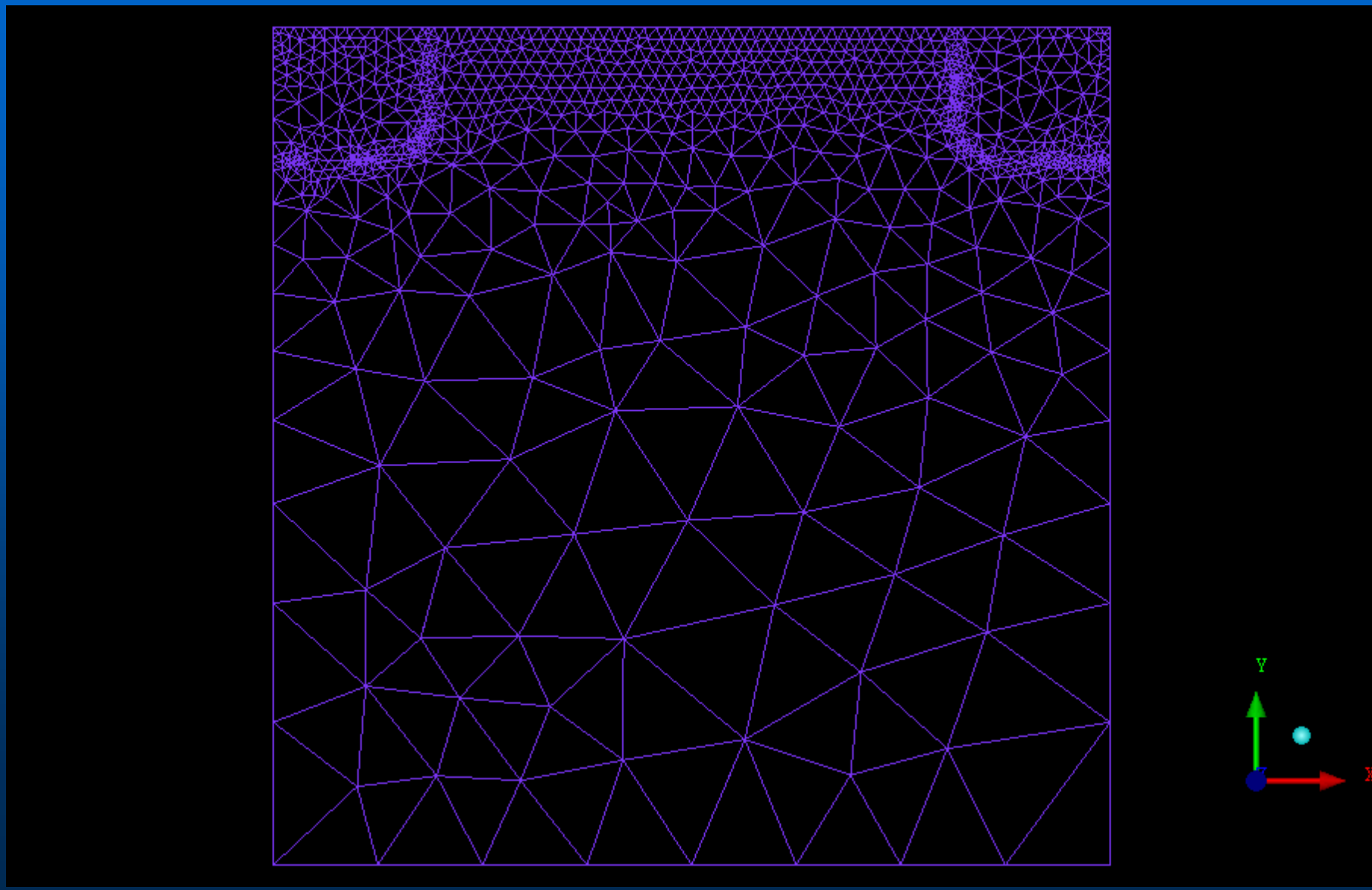
View CGNS file



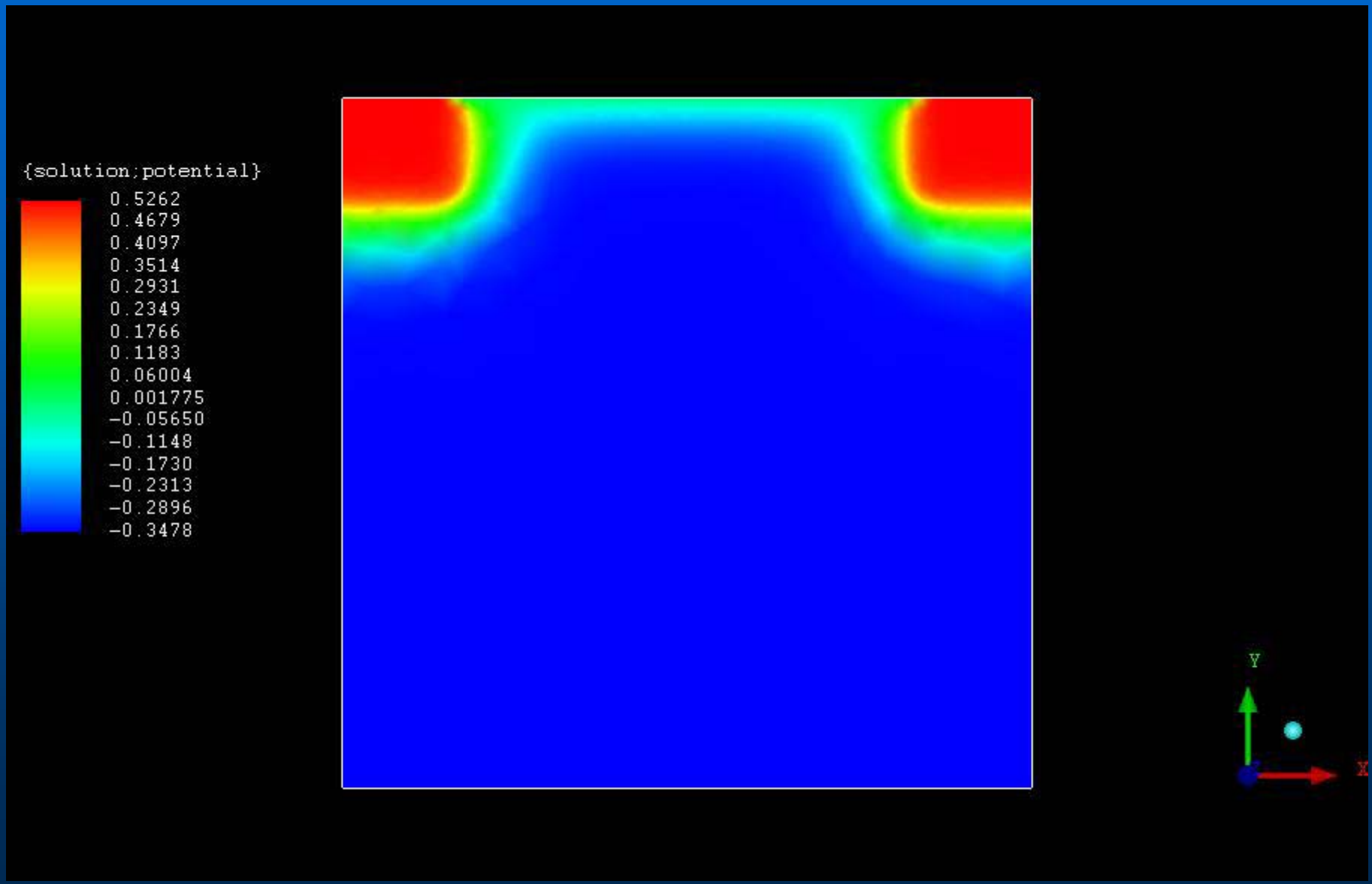
Show Mesh


CGNSplot : mos_sg.cgns
File
Display
Help
Base
SG_To_GSS
Zones
Si
SiO2
Visible
Color...
Regions
GridElements
WALL
OMSUB
ISGATE
OMDRAIN
OMSOURCE
Si_to_SiO2:conn
WALL:boco
OMSUB:boco
ISGATE:boco
Visible
Color...
Outline
Mesh
Flat
Smooth

Mesh editor (by ICEM)



Post process (by ICEM)



Command file

At present, lex and yacc are used to parse command file. It contains:

- Various run time parameters
- Boundary condition
- Voltage source attached to BC
- Solver specification

An example of command file 1

```
# common line
# All the command has the syntax: keyword  parameter=string|double
# String must begin with character or underline.
# Character, digital, underline and dot are allowed in string.
# The length of string is limited to 30 characters.
# Numerical value double support c syntax float point number.
# The Unit in command file.:
# Time Unit:ps  voltage Unit:V  Freq Unit:THz Length Unit:cm
#=====================================================================
# static command, in lower case
set MeshFile    = mesh.cgns # specify cgns file which contains mesh,doping and boundary lable
set ModelFile   = model.dat # specify physical model
set Carrier     = pn        # specify carrier type support p,n or pn
set DeviceDepth = 0.01      # device depth in Z dimension. Unit:cm
set LatticeTemp = 3e2       # specify initial temperature of device. Unit:K
set MeshScale   = 1e-6      # the internal unit of length during the simulation. For DDM
                            #  default value 1e-6cm is ok. For HDM 1e-4cm or 1e-5cm is more convenient.
#---------------------------------------------------------------------
# voltage source.
# ID can be specified by user, limited to 32 characters, c syntax.
# reference: voltage source in spice model
vsource Type = VDC    ID = GND  Tdelay=0 Vconst=0
vsource Type = VDC    ID = VCC  Tdelay=0 Vconst=5
vsource Type = VSIN   ID = Vs   Tdelay=1 Vamp=0.1 Freq=1e-6
vsource Type = VEXP   ID = V1   Tdelay=0 TRC=1 TFD=3 TFC=1 Vlo=0 Vhi=1
vsource Type = VPULSE ID = V2   Tdelay=0 Tr=1 Tf=1 Pw=5 Pr=10 Vlo=0 Vhi=1
```

An example of command file 2

```
#-----------------------------------------------------------------------
# specify boundary condition.
# ID must accord with the boundary name in cgns file
# electrode boundary
#       OhmicContract|SchottkyContract|GateContract|InsulatorContract
# they have parameters of parasite res,cap and ind  unit:Om,F,H
# SchottkyContract has an extra parameter,the barrier height.
# GateContract can specify the workfunc of gate electrode.
# InsulatorContract offers a simple way for describing Si/SiO2 interface,
#       the Thick of oxide must be specified,
# NeumannBoundary has heat transfer rate parameter Kapa. unit ?
# InsulatorInterface is the interface of Si/SiO2, has a fixed charge density QF,unit ?

boundary Type = InsulatorContract  ID = SiSiO2     Res=0 Cap=0 Ind=0 Thick=1e-6 QF=0
boundary Type = InsulatorInterface ID = IFACE      QF=0
boundary Type = GateContract       ID = GATE       Res=0 Cap=0 Ind=0 WorkFunction=0
boundary Type = NeumannBoundary    ID = WALL       Kapa=0
boundary Type = SchottkyContract   ID = sgate      Res=0 Cap=0 Ind=0 Vbarrier=-0.8
boundary Type = OhmicContract      ID = OMANODE    Res=0 Cap=0 Ind=0
boundary Type = OhmicContract      ID = OMCATHODE  Res=0 Cap=0 Ind=0
boundary Type = OhmicContract      ID = OMSOURCE   Res=0 Cap=0 Ind=0
boundary Type = OhmicContract      ID = OMDRAIN    Res=0 Cap=0 Ind=0
boundary Type = OhmicContract      ID = OMSUB      Res=0 Cap=0 Ind=0
#FloatMetalGate   may support later
```

An example of command file 3

```
#=============================================================================
# drive command, specify the solving process.
# keyword is in upper care
# reference: medici user's guide

#-------------------------------------------
# METHOD Type = DDM scheme = [Newton|Gummel] &
#        SNES = [LineSearchCubic|LineSearchQuadratic|LineSearchNo|TrustRegion]  &
#        TStep = time_number
# METHOD Type = HDM scheme = [Implicit|Explicit] FluxFunc = [AUSM|Roe]  &
#        Reconstruct = [FirstOrder|SecondOrder] CFL = cfl_number
#-------------------------------------------
# ATTACH Electrode = electrode_name  VApp = vsource_name
#-------------------------------------------
# SOLVE Type = EQUILIBRIUM
# SOLVE Type = STEADYSTATE
# SOLVE Type = DCSWEEP  VScan = electrode_name  IVRecord = electrode_name  &
#       IVFile = file_name VStart = v_number VStep = v_number VStop = v_number
# SOLVE Type = TRANSIENT   AUTOSAVE = time_number   IVRecord = electrode_name &
#       IVFile = file_name TStart = time_number    TStop = time_number
#-------------------------------------------
# MODELS not supported yet
#-------------------------------------------
# IMPORT CoreFile = file_name
#-------------------------------------------
# EXTRACT CoreFile = file_name    AscFile = file_name
#-------------------------------------------
# REFINE Variable = [Doping|Potential] Measure = [Linear|SignedLog] Dispersion = number
#-------------------------------------------
# PLOT   Variable = Mesh Resolution=[Low|Middle|High] PSFile=file_name
# PLOT   Variable =[Na|Nd|ElecDensity|HoleDensity|Potential|EFieldX|EFieldy|Temperature]
#        Resolution=[Low|Middle|High] PSFile=file_name Measure=[Linear|SignedLog]
#        AzAngle=angle_number ElAngle=angle_number Style=[Scale|Color|GrayLevel]
#=============================================================================
```

An example of command file 4

```
METHOD    Type = DDM    Scheme = Newton    TStep=1e3          #specify DDM method,this is the default solver.
PLOT      Variable=Mesh
REFINE    Variable=Doping Measure=SignedLog Dispersion=1    #refine by doping
SOLVE     Type=EQUILIBRIUM                                  #compute equilibrium state
REFINE    Variable=Potential Measure=Linear Dispersion=0.1 #refine by potential
PLOT      Variable=Mesh
SOLVE     Type=EQUILIBRIUM                                  #compute equilibrium state again
PLOT      Variable=Na Resolution=Middle     AzAngle=240  ElAngle=40  Style=Scale
PLOT      Variable=Nd Resolution=Middle     AzAngle=240  ElAngle=40  Style=Scale
PLOT      Variable=ElecDensity PSFile=electron Resolution=High     AzAngle=240  ElAngle=40  Style=Color
PLOT      Variable=HoleDensity PSFile=hole     Resolution=High     AzAngle=240  ElAngle=40  Style=Color
PLOT      Variable=Potential   Resolution=High     AzAngle=240  ElAngle=40  Style=GrayLevel
# extract mesh and solution
EXTRACT   CoreFile=init.cgns

IMPORT    CoreFile=init.cgns                      # import it
ATTACH    Electrode=OMCATHODE  VApp=GND          # attach vsource to boundary(electrode)
# DC sweep
SOLVE     Type=DCSWEEP  VScan=OMANODE  IVRecord=OMANODE  IVFile=iv.txt VStart=0 VStep=1e-2 VStop=1

#IMPORT CoreFile=break.cgns
ATTACH    Electrode=OMANODE     VApp=VCC          VApp=Vs
# specify HDM method
METHOD    Type=HDM              Scheme=Explicit  FluxFunc=AUSM  Reconstruct=FirstOrder CFL=0.1
SOLVE     Type=TRANSIENT        AUTOSAVE=1    TStart=0     TStop=10

METHOD    Type=HDM              Scheme=Implicit  FluxFunc=AUSM  Reconstruct=SecondOrder CFL=2
SOLVE     Type=TRANSIENT        AUTOSAVE=1    TStart=10     TStop=20
```

Graphic plot

- GSS requests Xwindow to do graphic plot.
- Support both 2D mesh displaying and 3D plotting of results.
- User can choose style, color and change view angle by mouse.
- In the future, Graphic user's interface will be built.

Solvers

- The two popular methods in semiconductor simulation—DDM and HDM—are supported both.
- Use DDM method to get a zero bias solution is very fast and accurate while HDM needs a lot of time to get convergence.
- HDM is suit for sub-micron device simulation such as MESFET and HEMPT

DDM Level 1

DDM is the basic method for semiconductor simulation.It employs Newton's iterative method to solve nonlinear equations.

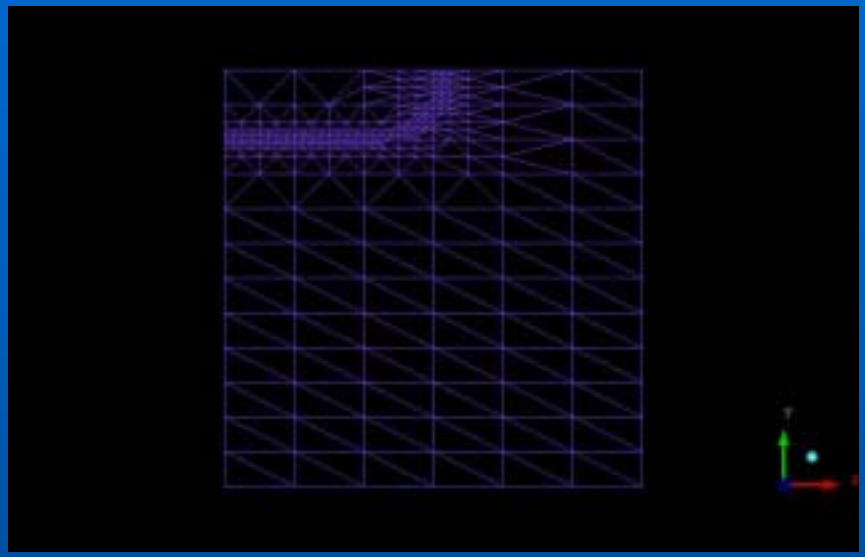
With the help of PETSC, the DDM solver is ready to go. Because PETSC support line search and trust region method, GSS can get convergence in most of the situation.

DDM Level 2

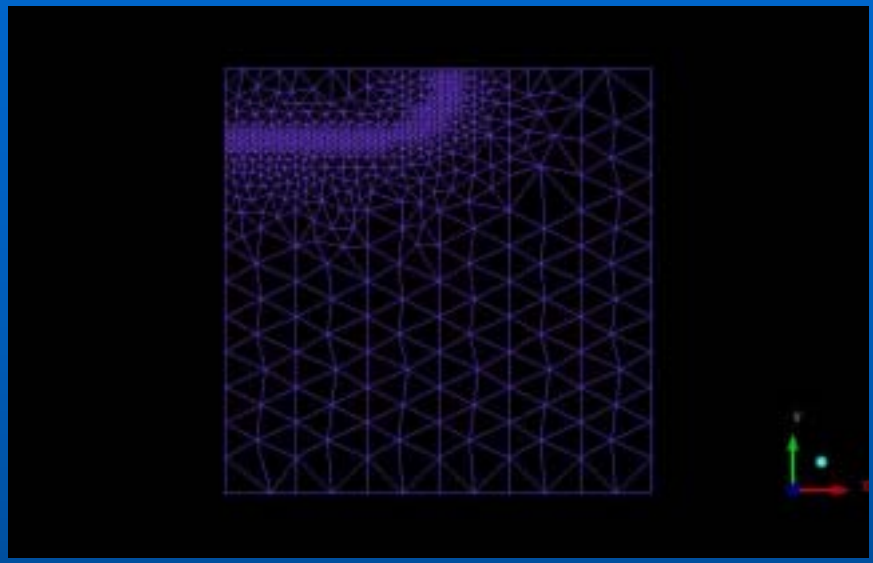
The thermal effects are critical when device attacked by HPM. Beyond the basic DDM solver, a lattice temperature corrected DDM solver (L2) is developed, which can simulate the thermal phenomena of devices.

Unfortunately, DDM L2 runs 2-10 times slower than original edition.

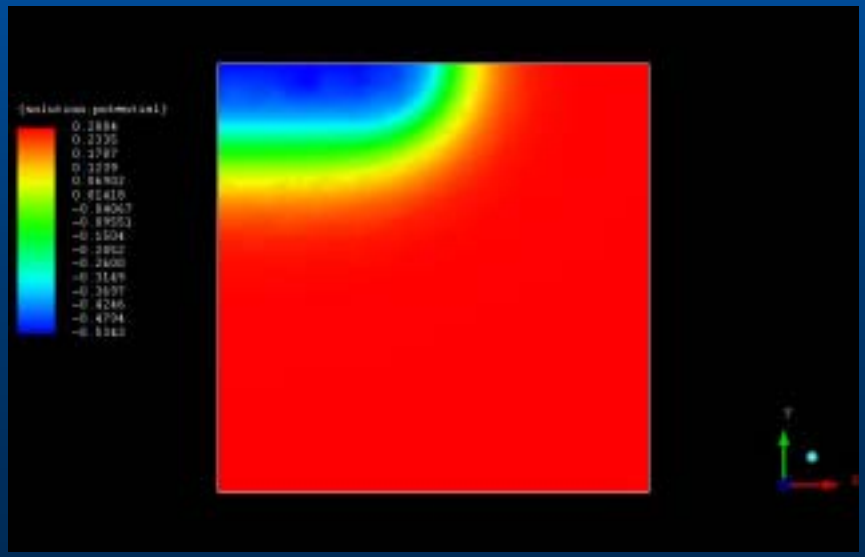
Example 1 PN diode



Original mesh

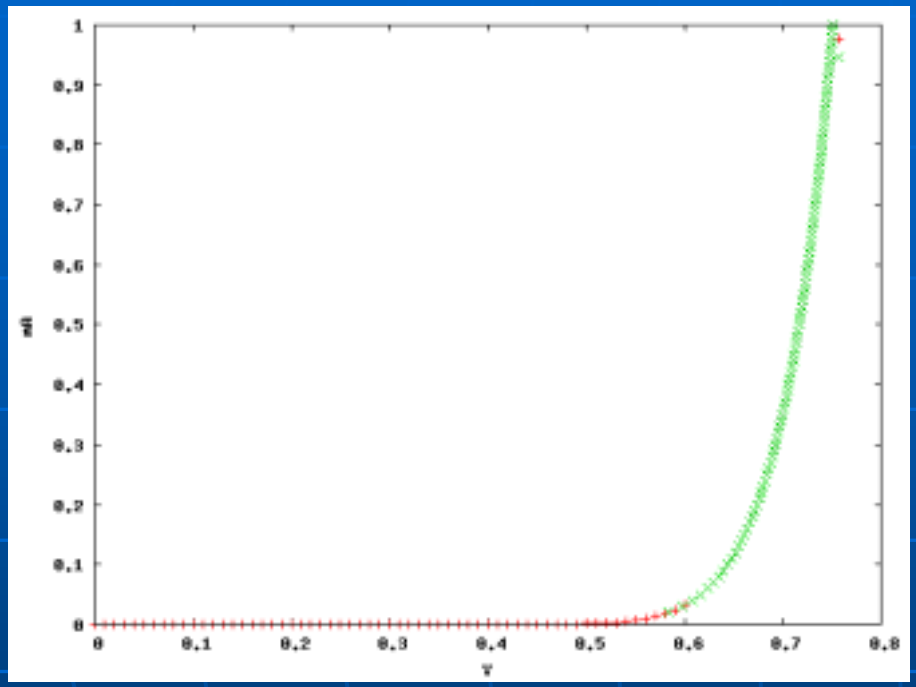


Refined mesh



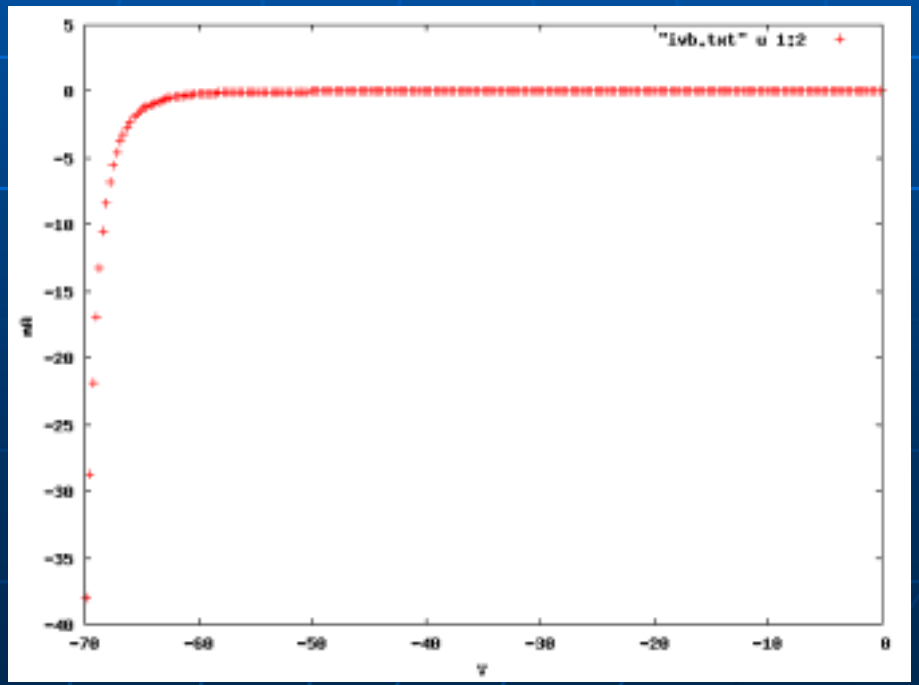
Potential distribution of equilibrium

Example 1 PN diode

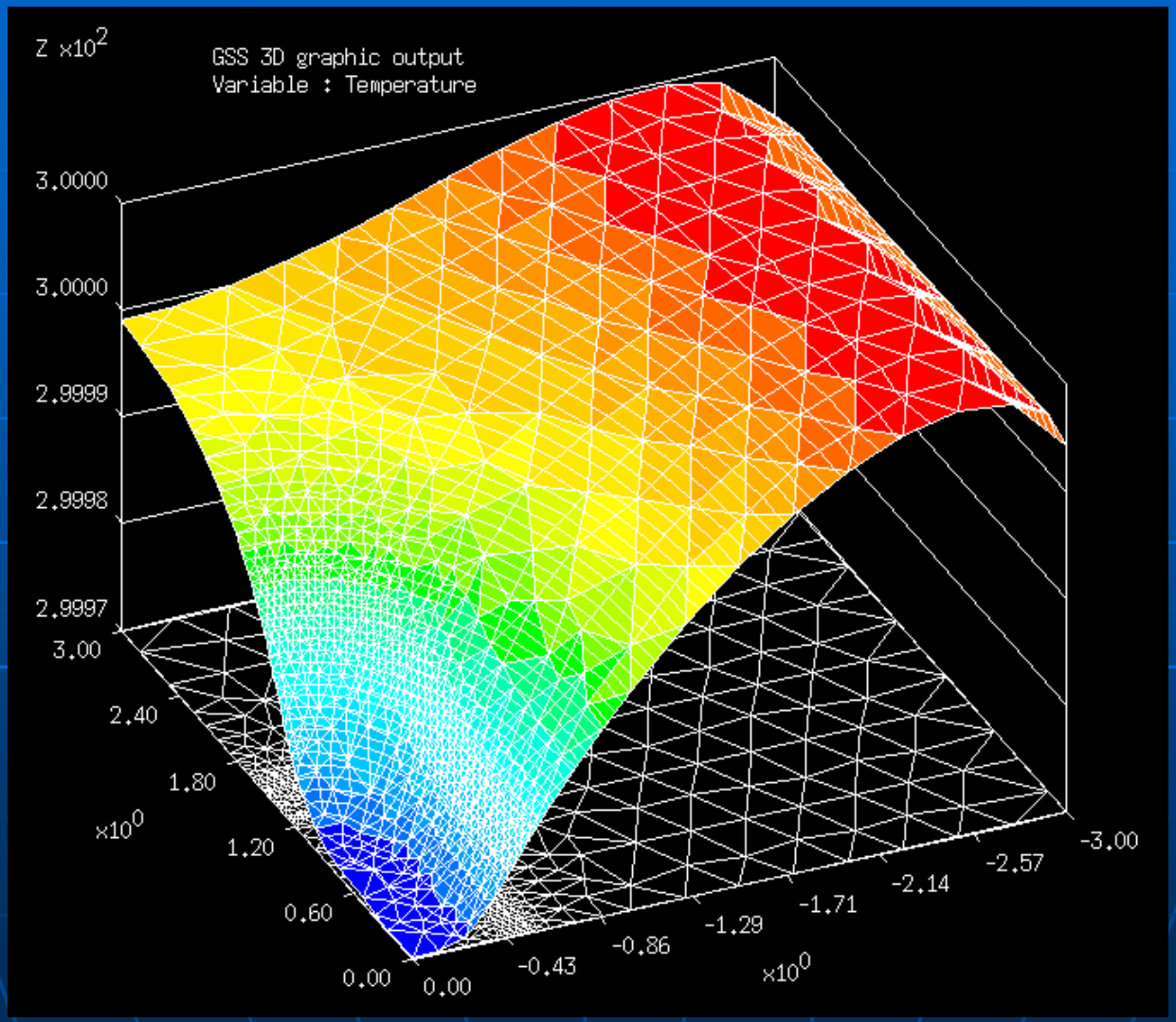


IV: forward

IV: backward

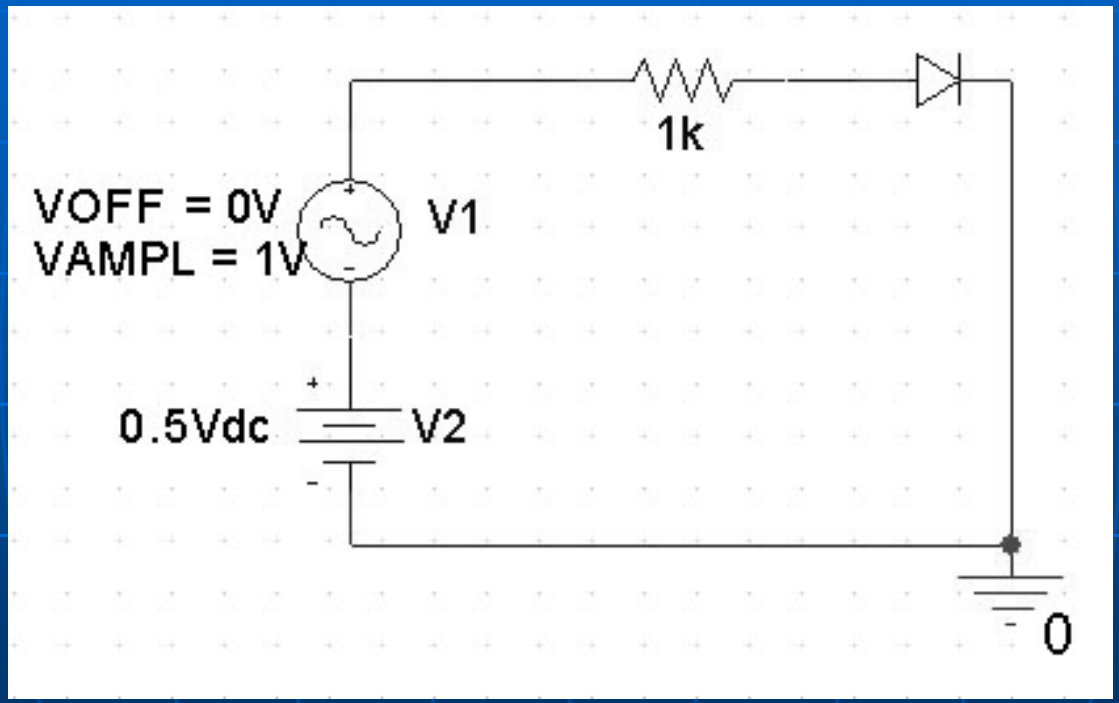


Example 1 PN diode



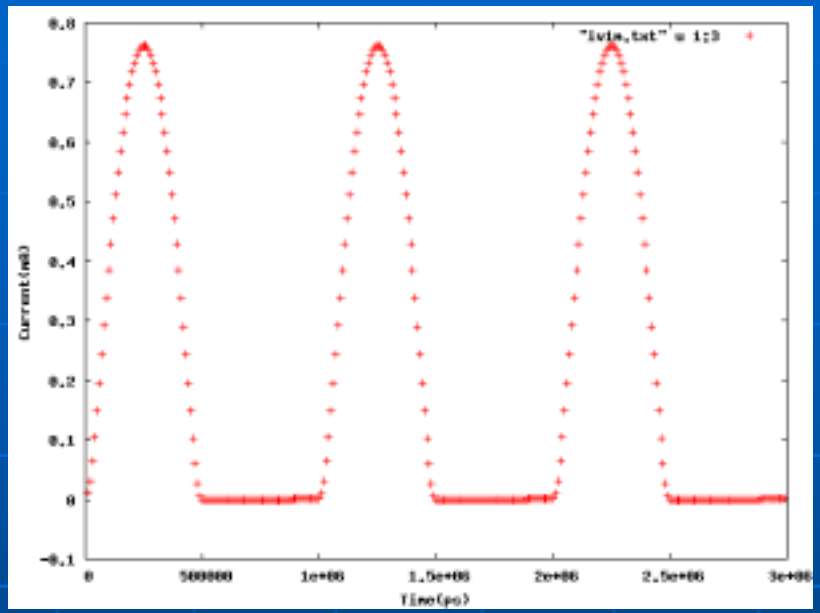
Temperature Distribution of forward bias

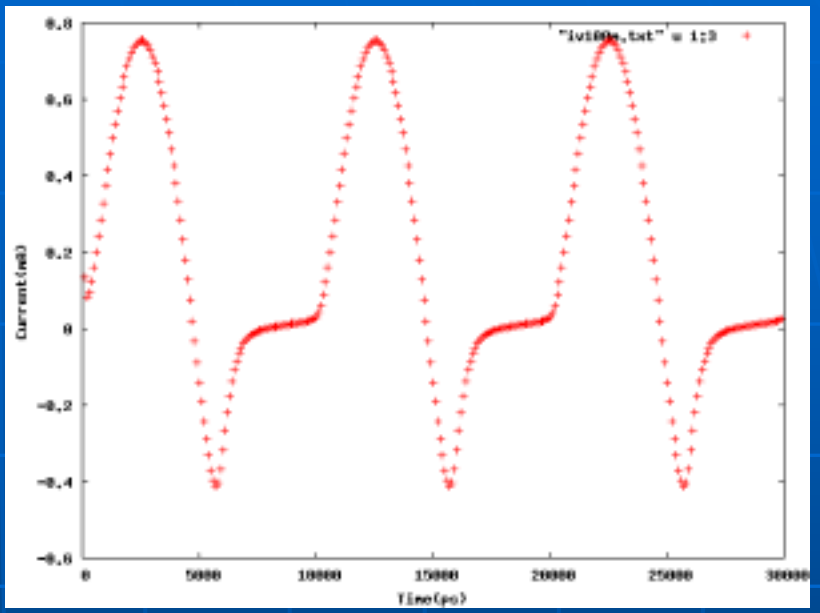
Example 1 PN diode

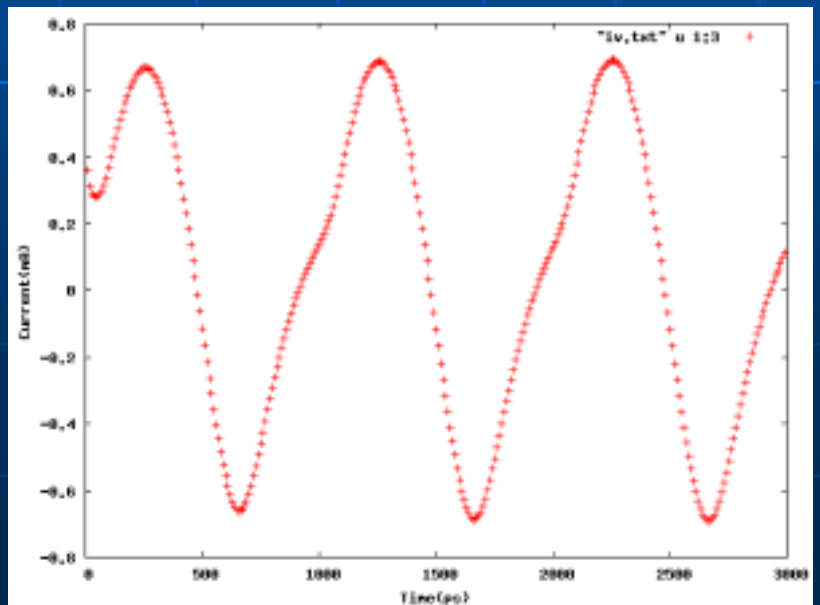


Frequent dependent simulation

Example 1 PN diode

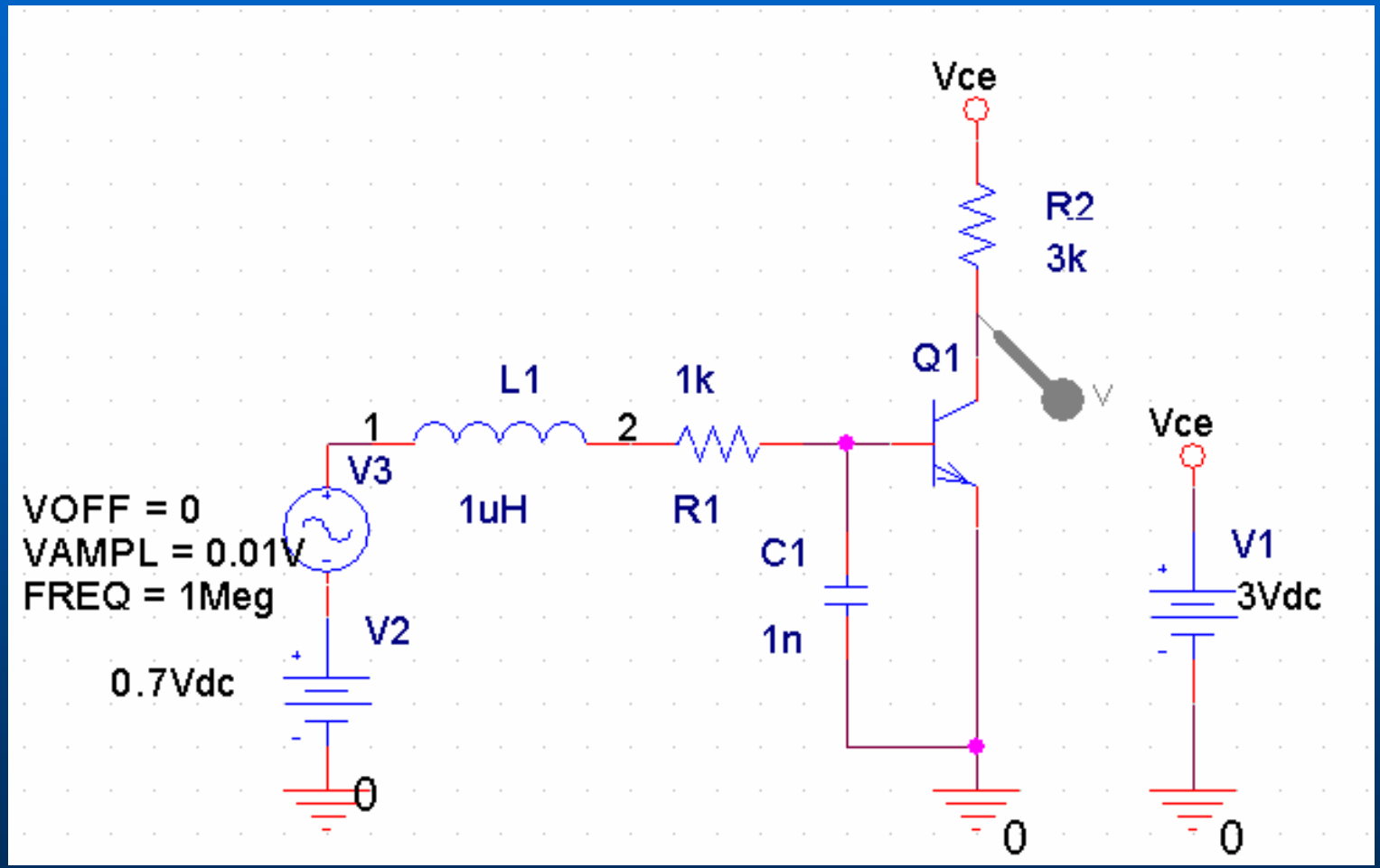






Top left: 1MHz
Top right: 100MHz
Bottom left: 1GHz

Example 2 BJT circuit

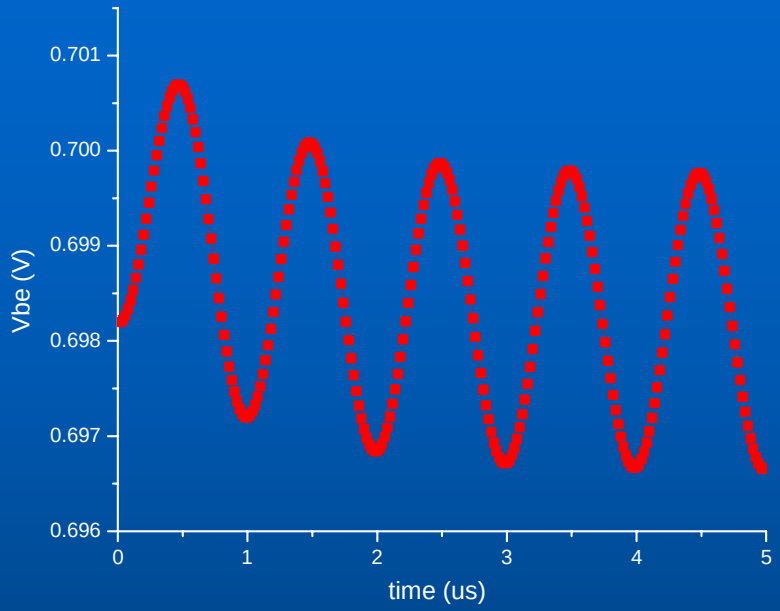


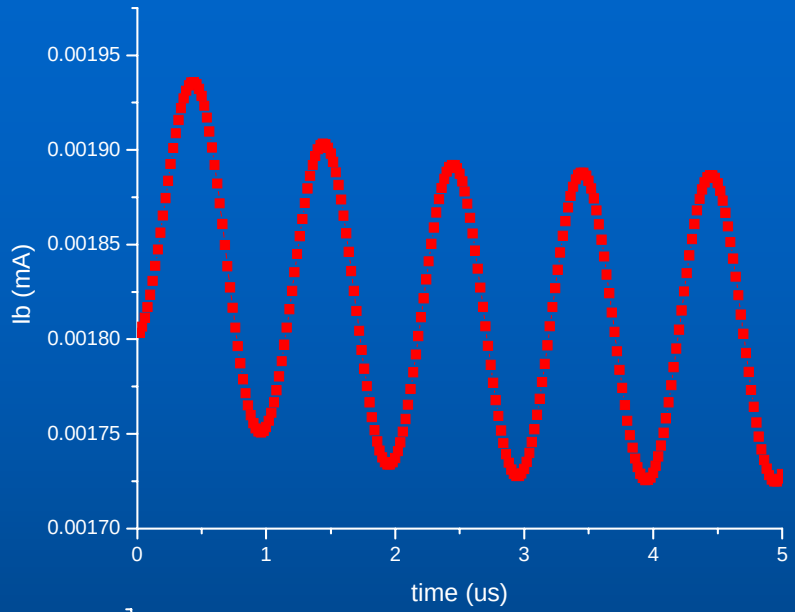
Circuit scheme

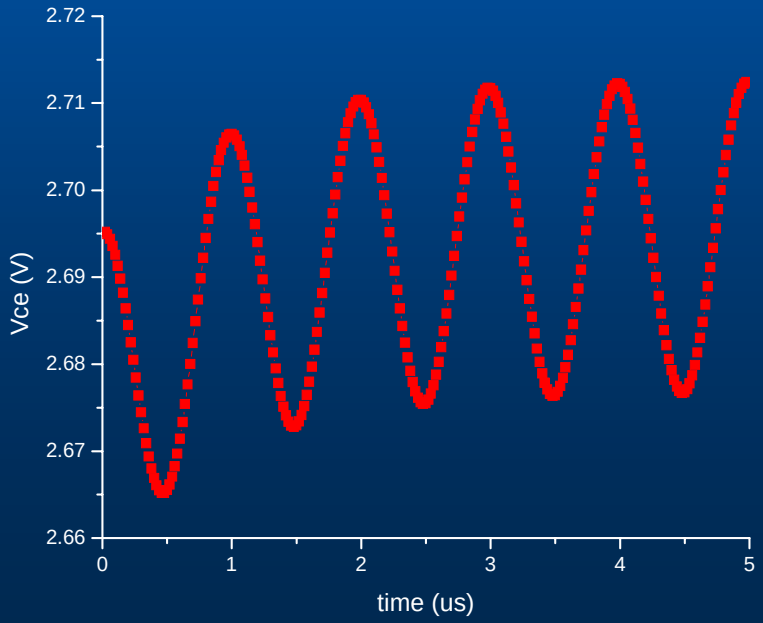
Input file

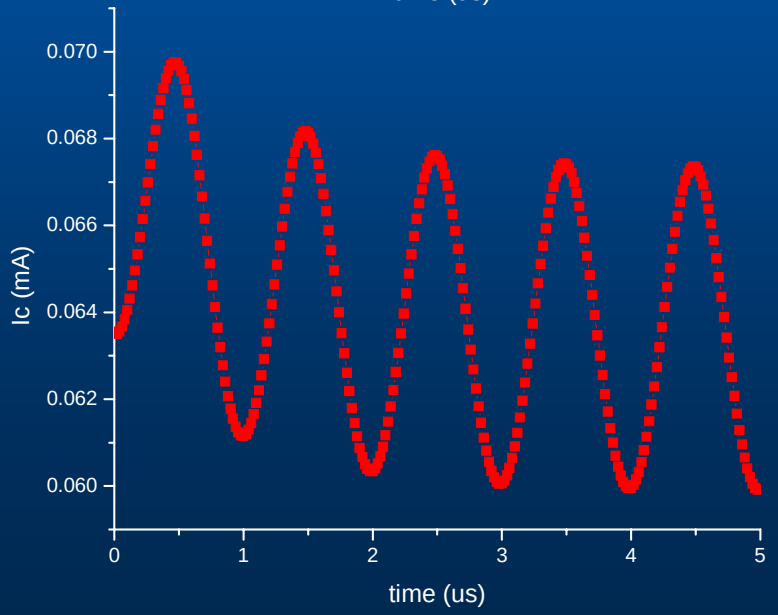
```
# advanced topic:we will compute transient reaction of a npn Transistor
#=====================================================================
set Carrier     = pn        # specify carrier type support p,n or pn
set DeviceDepth = 0.01      # device depth in Z dimension. Unit:cm
set LatticeTemp = 3e2       # specify initial temperature of device. Unit:K
set DopingScale = 1e16
#---------------------------------------------------------------------
# voltage source and current source is needed here.
vsource Type = VDC    ID = GND  Tdelay=0 Vconst=0
vsource Type = VDC    ID = VCC  Tdelay=0 Vconst=3.0
vsource Type = VDC    ID = Vb   Tdelay=0 Vconst=0.7
vsource Type = VSIN   ID = Vs   Tdelay=0 Vamp=1e-3 Freq=1e6  # 1MHz
isource Type = IDC    ID = Ib   Tdelay=0 Iconst=0.002
isource Type = ISIN   ID = Is   Tdelay=0 Iamp=1e-5 Freq=1e6  # 1MHz
#---------------------------------------------------------------------
# specify boundary condition.
boundary Type = OhmicContact    ID=Base      Res=1000 Cap=1e-9 Ind=1e-6
boundary Type = OhmicContact    ID=Emitter   Res=0    Cap=0 Ind=0
boundary Type = OhmicContact    ID=Collector Res=3000 Cap=0 Ind=0
#=====================================================================
IMPORT   CoreFile=bias.cgns                          # import  result computed in step 2
METHOD   Type = DDML1   Scheme = Newton  NS=LineSearch LS=BCGS
ATTACH   Electrode=Collector    VApp=VCC
ATTACH   Electrode=Emitter      VApp=GND
#ATTACH   Electrode=Base        VApp=Vb  VApp=Vs             # use voltage boundary condition.
ATTACH   Electrode=Base     Type=Current IApp=Ib   IApp=Is   # use current boundary condition.
SOLVE    Type=STEADYSTATE                           # compute STEADYSTATE to get a initial value
METHOD   Type = DDML1   Scheme = Newton  NS=LineSearch LS=BCGS
SOLVE    Type = TRANSIENT  IVRecord=Collector  IVRecord=Base   \
         IVFile=iv.txt TStart=0 TStep=1e-8  TStop = 2e-6
```

Result of Transient simulation





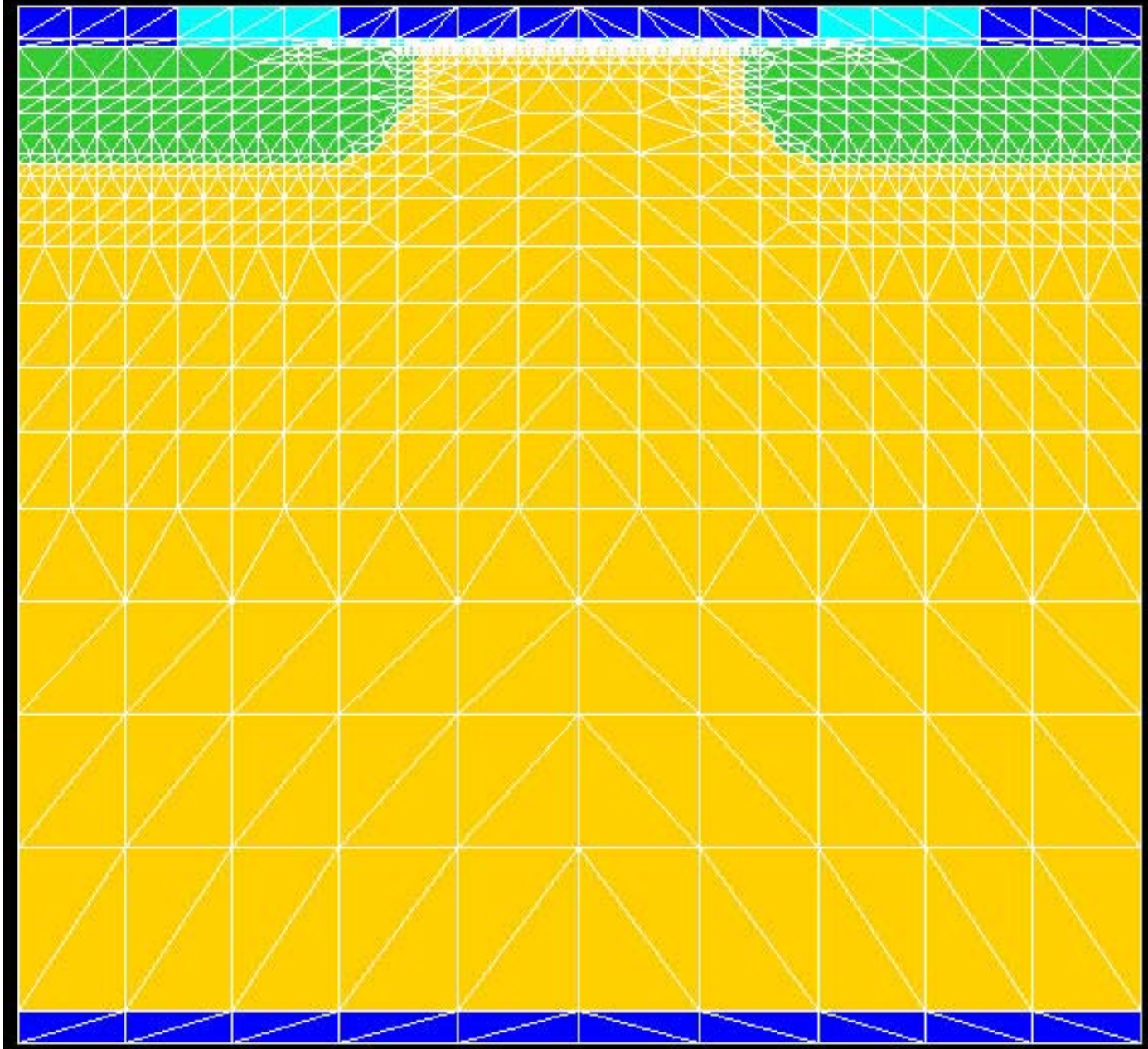




Example 3 multi-region NMOS

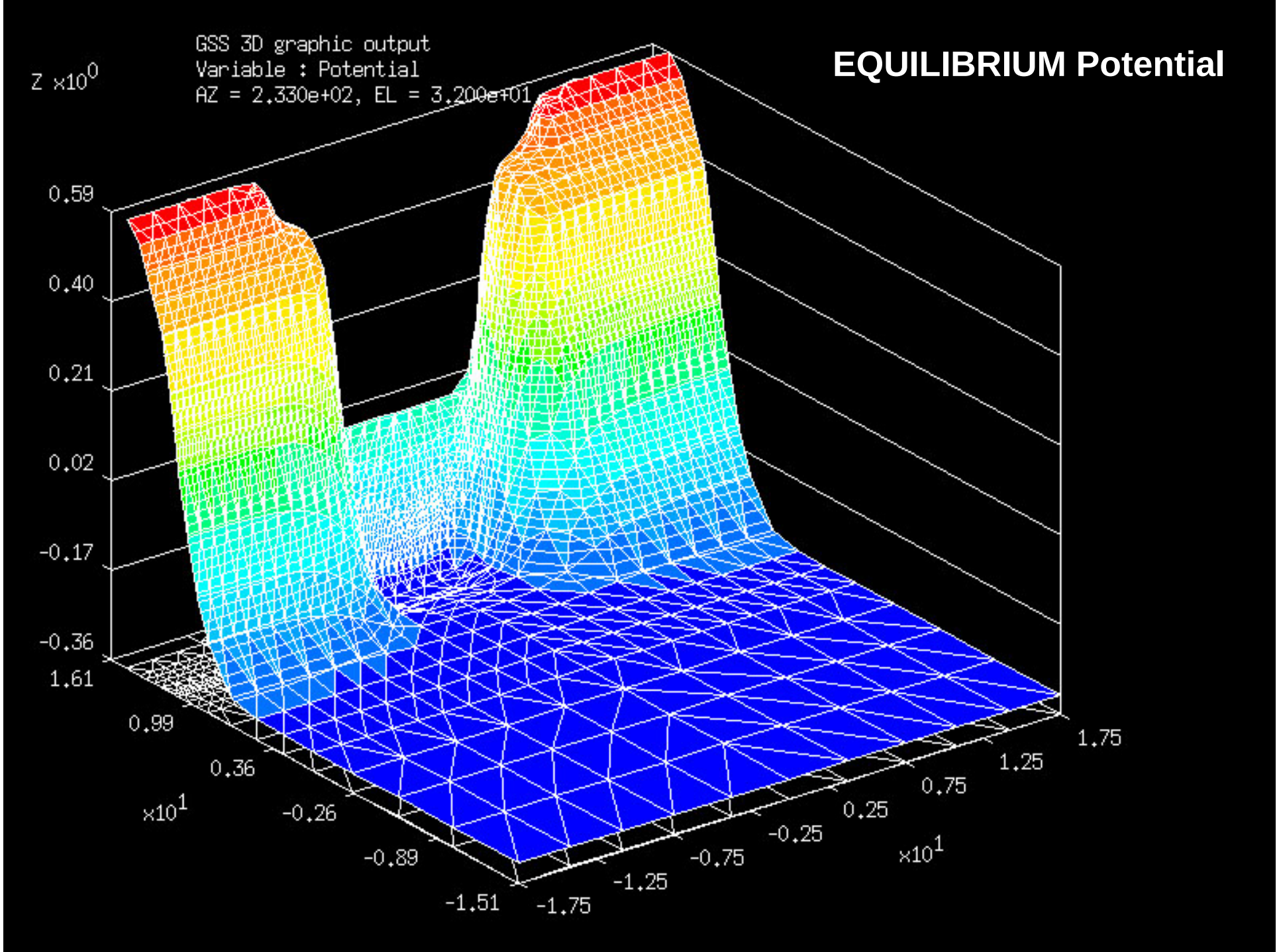
Here, a very complex NMOS transistor is simulated, which shows the multi-region processing capacity of GSS.

Mesh Structure

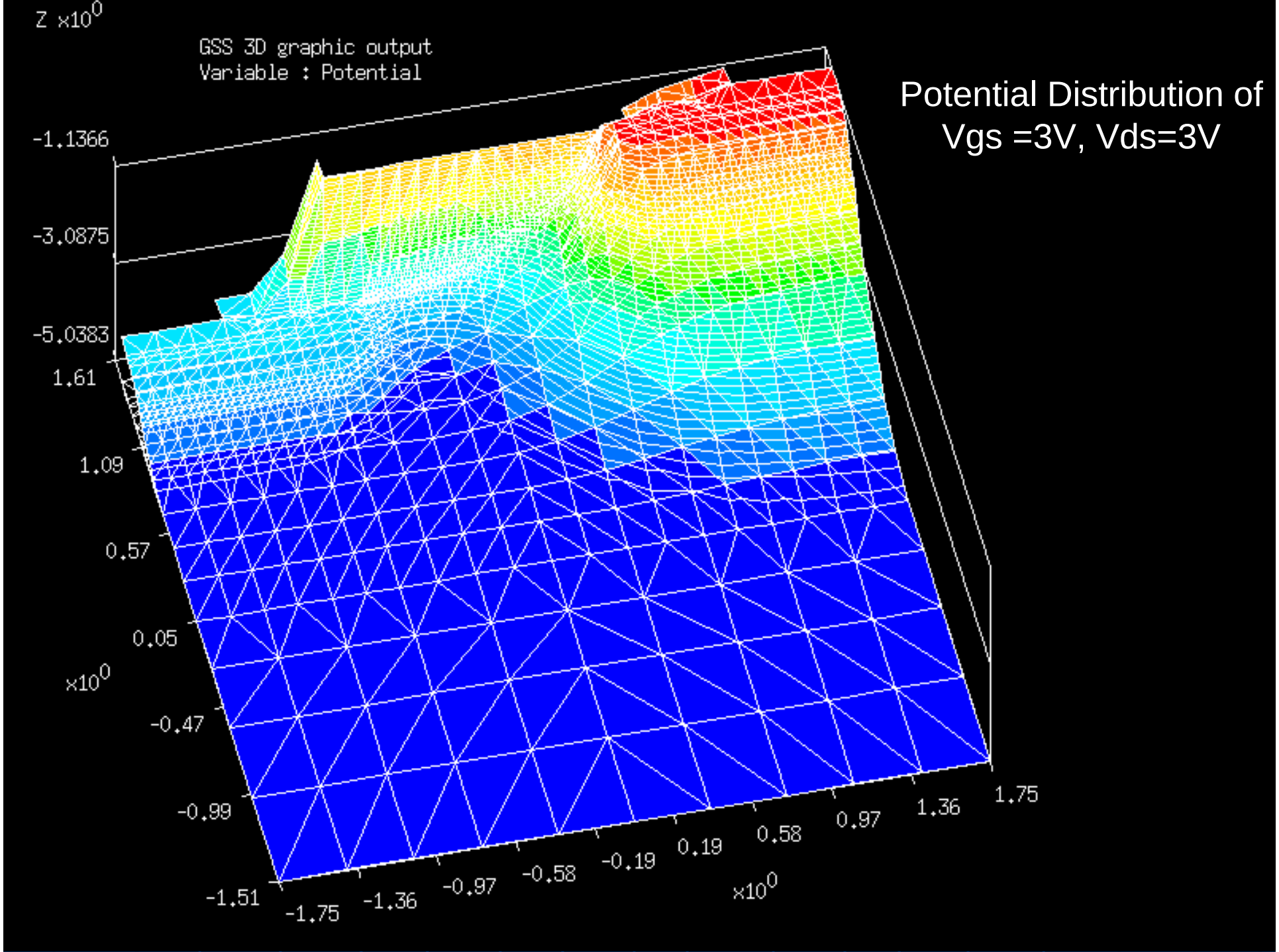


zone 1 lable : Drain
12 nodes
12 elements
23 voronoi edges
zone 2 lable : Gate
26 nodes
24 elements
49 voronoi edges
zone 3 lable : Oxide
146 nodes
200 elements
345 voronoi edges
zone 4 lable : Silicon
797 nodes
1478 elements
2274 voronoi edges
zone 5 lable : Source
12 nodes
12 elements
23 voronoi edges
zone 6 lable : Substrate
22 nodes
20 elements
41 voronoi edges

Press ESC to continue

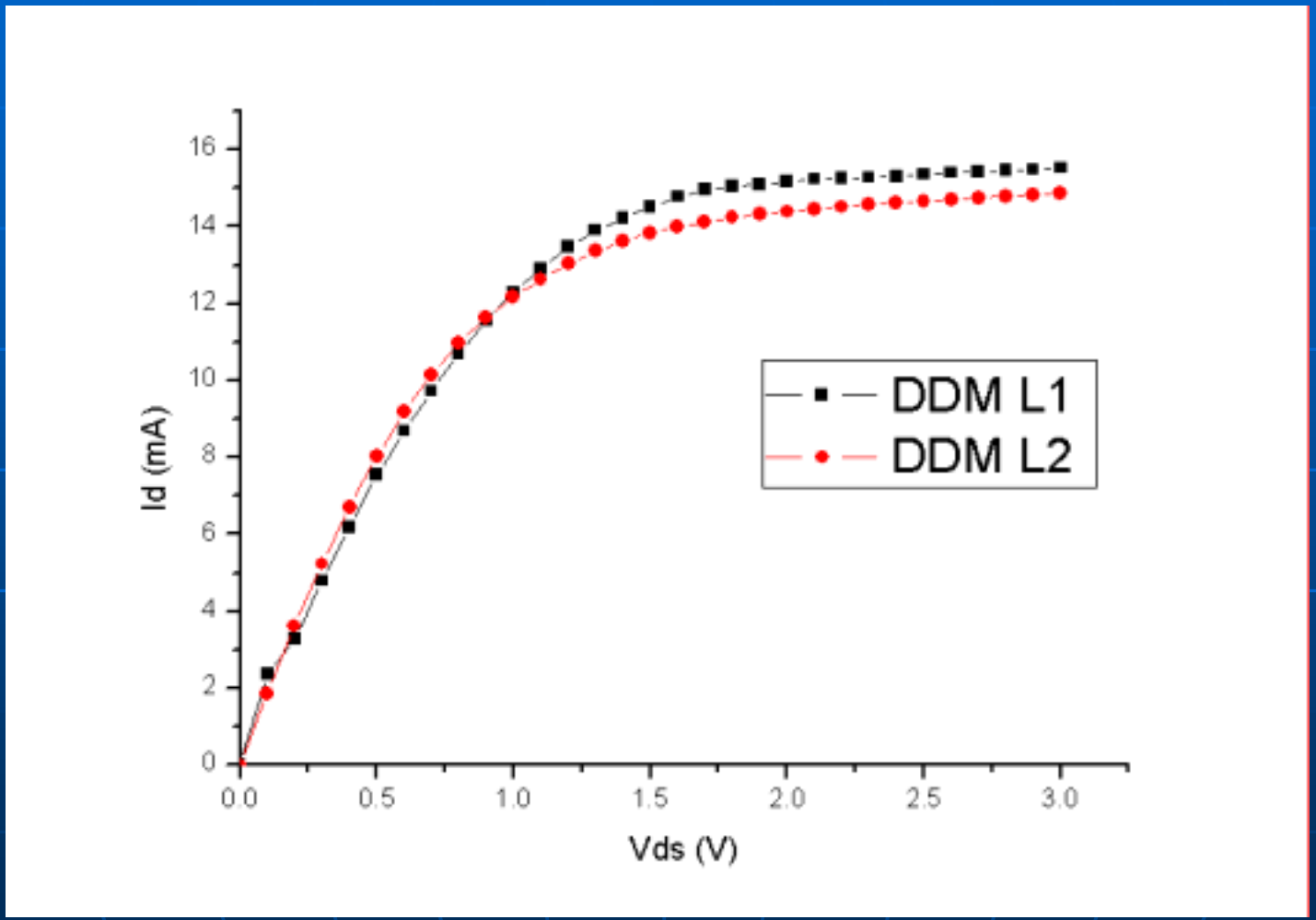


EQUILIBRIUM Potential



Potential Distribution of
Vgs =3V, Vds=3V

Example 3 multi-region NMOS



IV curve of Vgs=3V

HDM

- At present, both explicit and implicit HDM method are ok. The Roe and AUSM schemes are supported.

- The HDM solver is consisted of two main parts: a Poisson solver and a CFD solver. Optimization of Poisson

  and CFD solver is a future project.

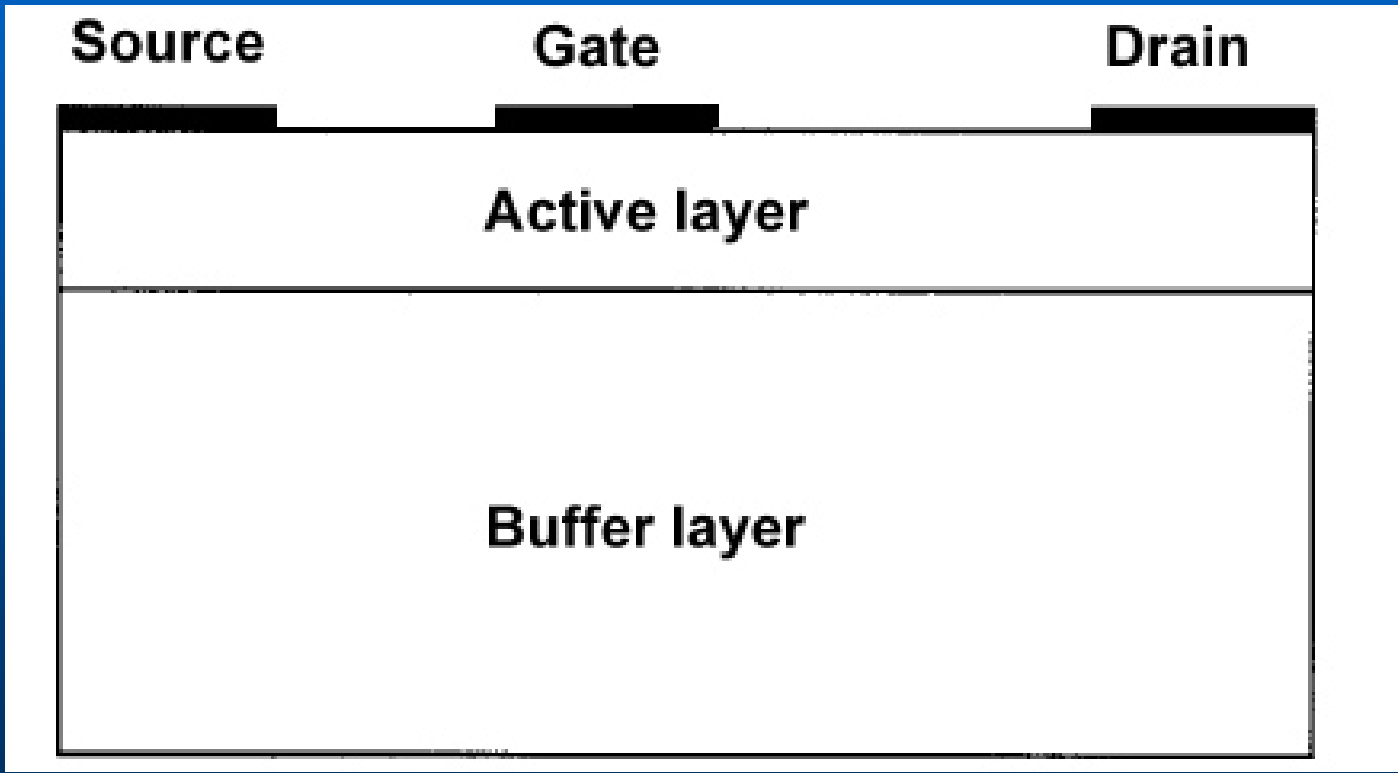
limitation of HDM

I did several tests with some transistors. The Numerical viscosity may cause terrible problems in bipolar transistor. HDM works well only with single carrier transistors such as MESFET and HEMT.

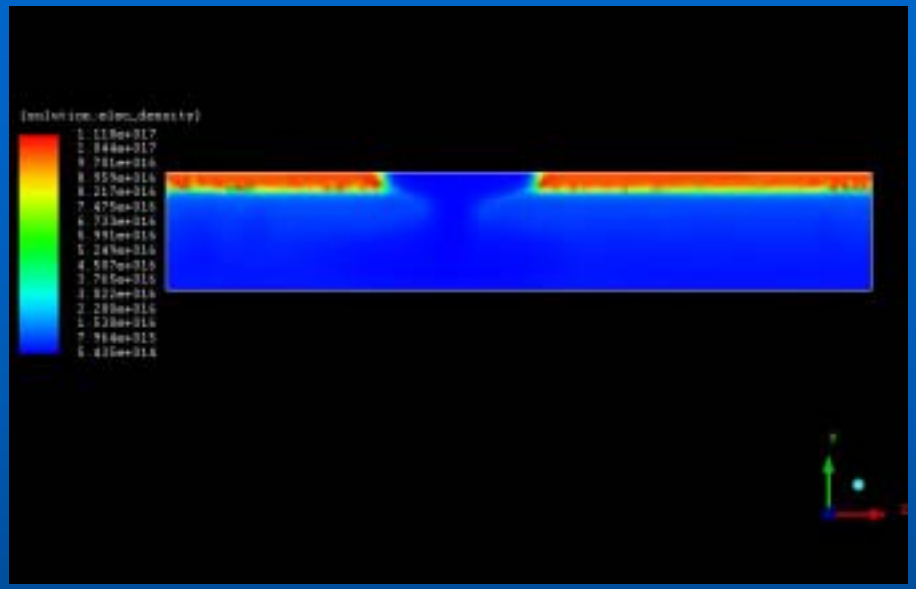
As a result, adaptive mesh refinement and second order reconstruction in space are done for anti-viscosity.

In the future, higher order Discrete Galerkin method may be introduced into GSS.

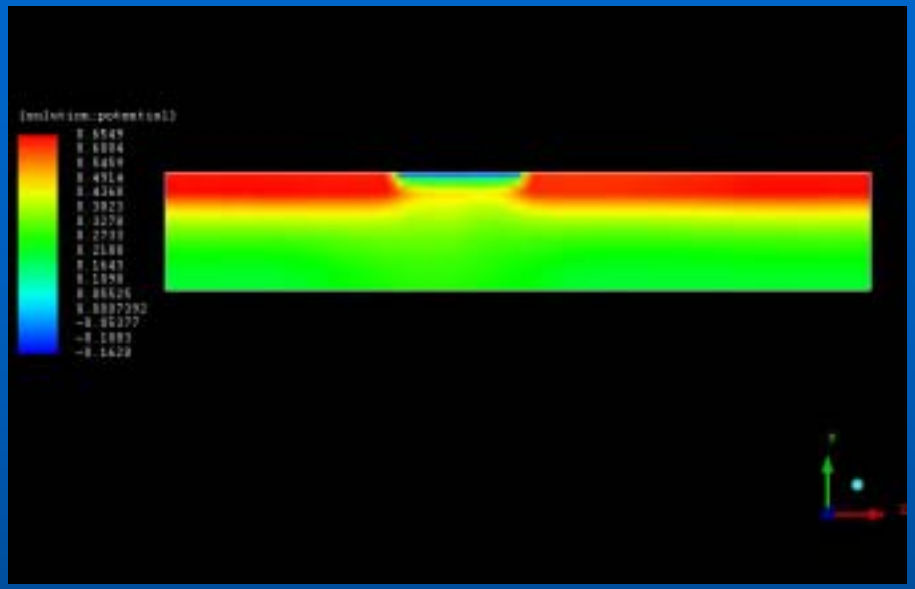
HDM example GaAs MESFET



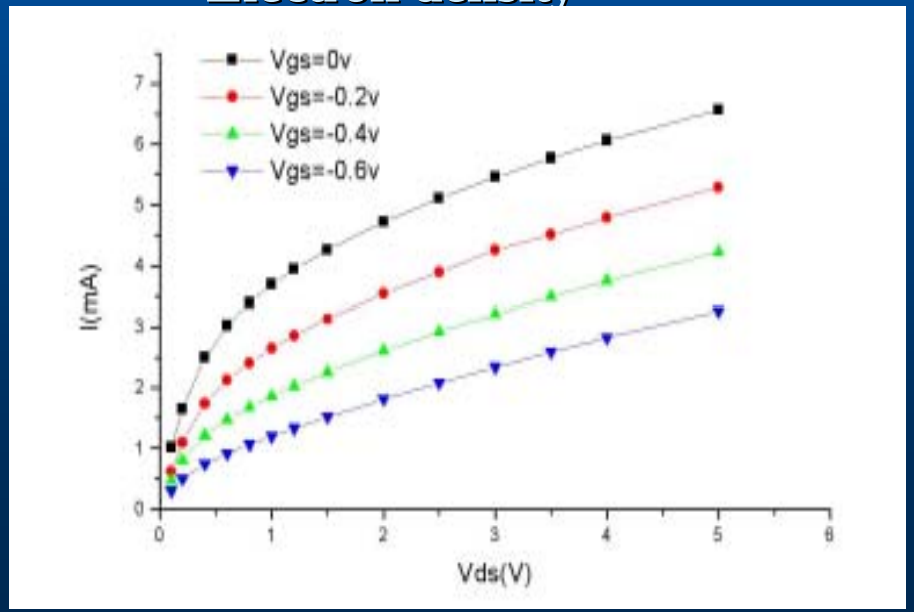
Result of MESFET



Electron density



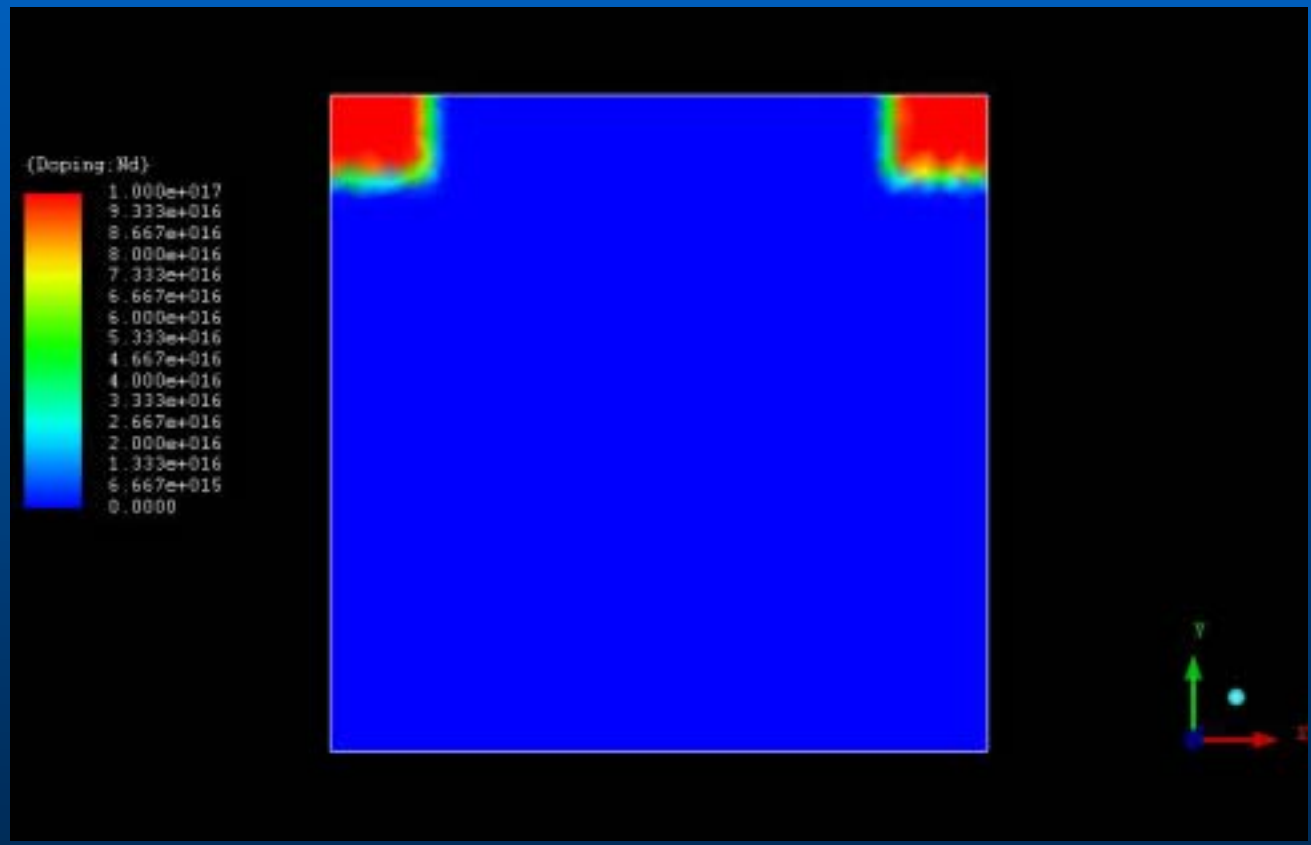
Potential



IV curvy

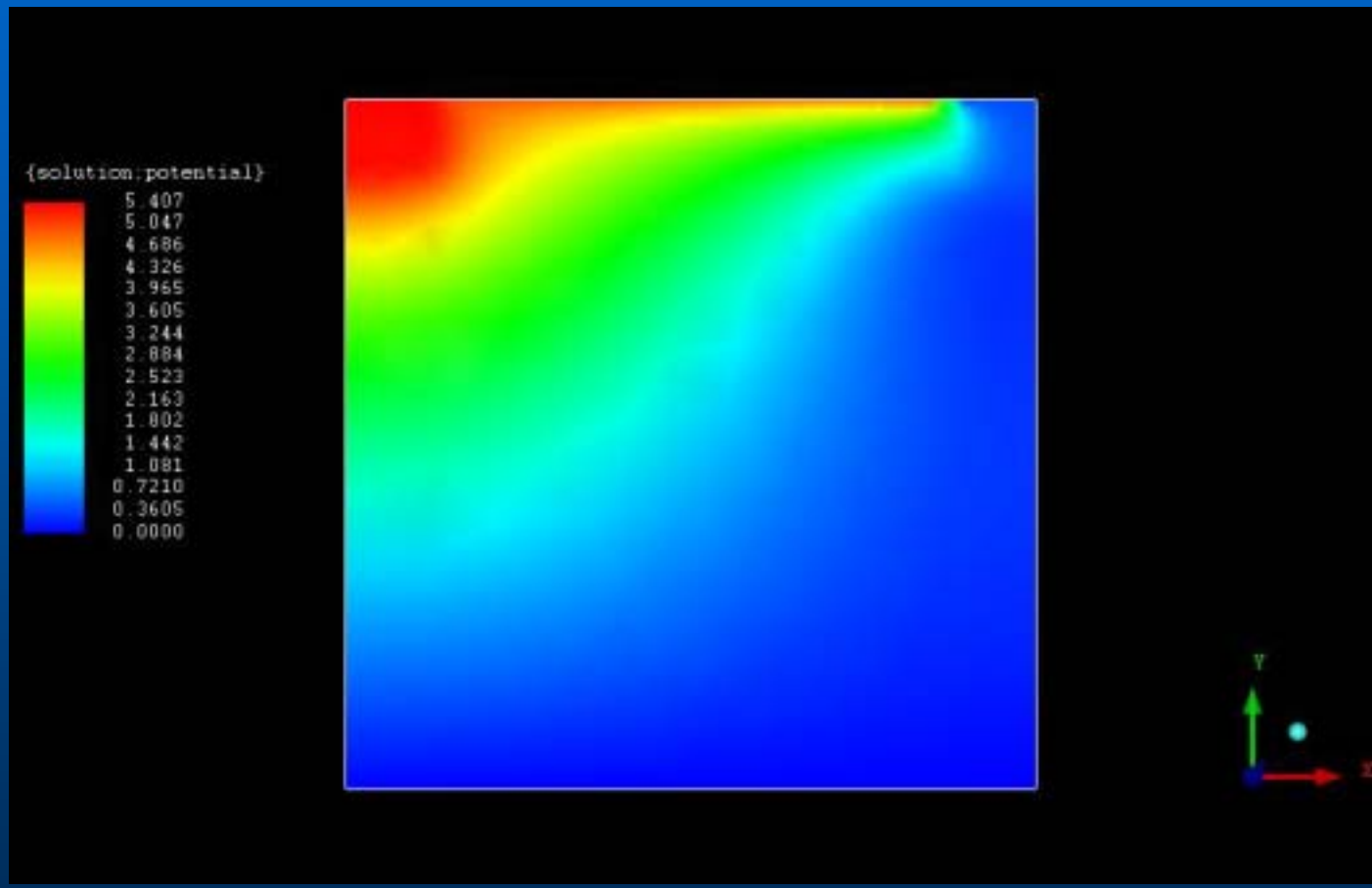
HDM example NMOS

Simplified NMOS Model. Na is set to zero. Only electron was considered.



Doping :Nd

Result of NMOS under Vds=5V, Vgs=5V



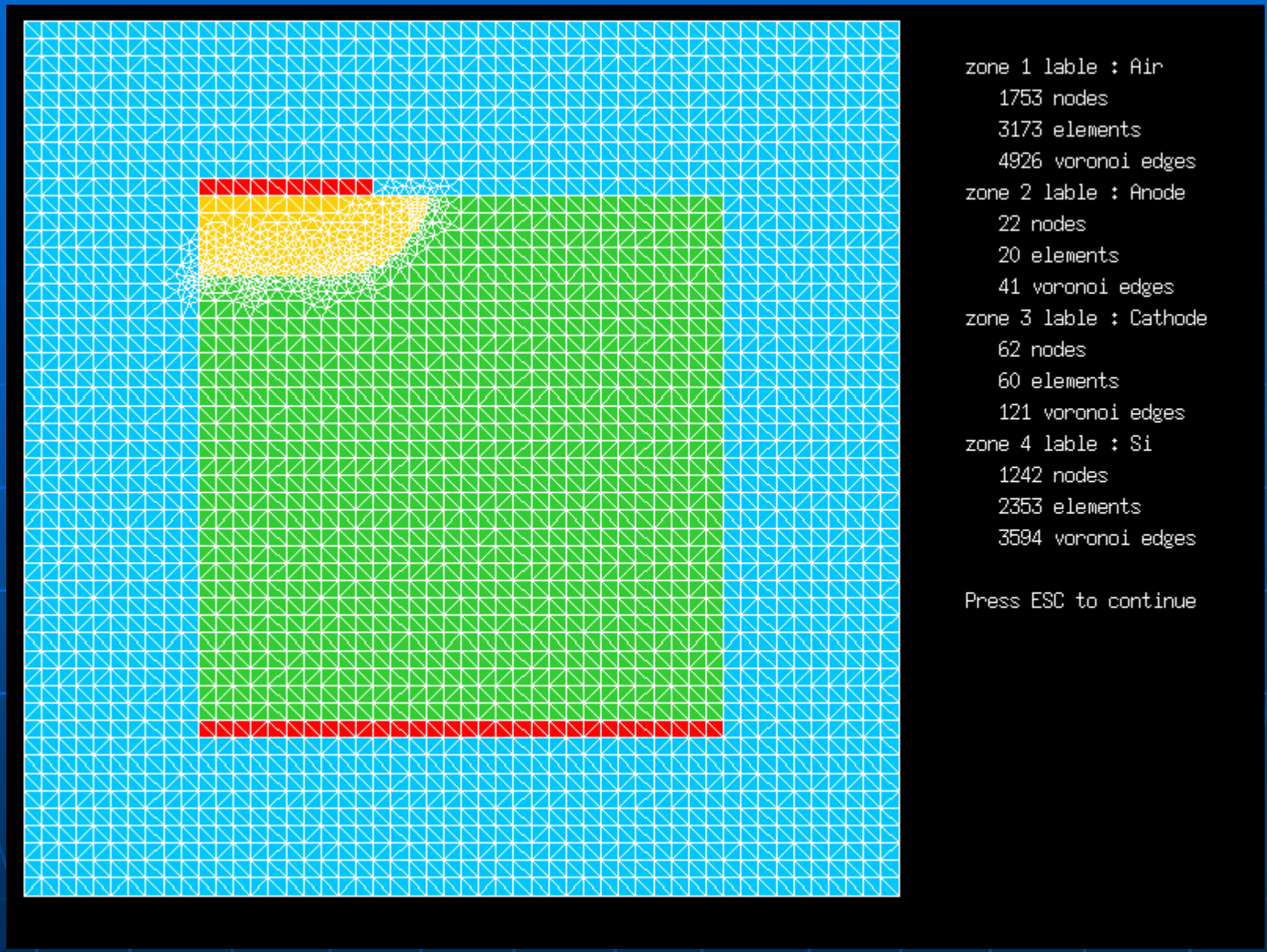
Potential

Future Works

- Build user-friendly Graphic User's Interface.
- support heterojunction device.
- Support optical mechanism.

*Support of FEM

For meeting the challenge of CCD simulation, GSS had introduced a background mesh, which enables using FEM to solve electromagnetic problems.



A PN diode with background mesh

*Run GSS on a Cluster

- Since GSS support multi-region mesh, to write a parallel edition of GSS is not so difficult.
- The nonlinear solver itself is well designed for cluster.

Special Thanks for Your Attention!

Presented by Gong Ding