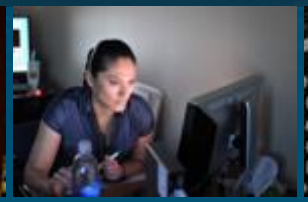
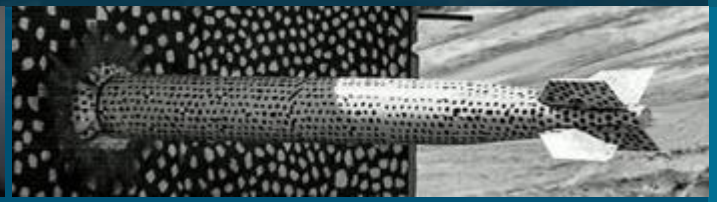
Charon: Basic Introduction Training





Lawrence C Musson, Gary Hennigan, Xujiao Gao, Mihai Negoita, and Andy Huang

Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2020-14169 PE

What does Charon do?

- Drift-Diffusion PDE solver for modeling charge carrier flow

Electric Potential

$$\begin{cases} \nabla \cdot \left(\epsilon \vec{\mathbf{E}}\right) = q\left(p - n + C\right) \\ \vec{\mathbf{E}} = -\nabla V \end{cases}$$

Carrier Transport Equations

$$\begin{aligned} \vec{\mathbf{J}}_n &= q\left(n\mu_n \vec{\mathbf{E}} + D_n \nabla n\right) \\ \vec{\mathbf{J}}_p &= q\left(p\mu_p \vec{\mathbf{E}} - D_p \nabla p\right) \end{aligned}$$

Conservation

$$\begin{aligned} \nabla \cdot \vec{\mathbf{J}}_n - qR &= q\frac{\partial n}{\partial t} \\ -\nabla \cdot \vec{\mathbf{J}}_p - qR &= q\frac{\partial p}{\partial t} \end{aligned}$$

Lattice Heating

$$\nabla \cdot (\kappa \nabla T_L) + H = \rho c \frac{\partial T_L}{\partial T}$$

TCAD code for modeling semiconductor performance including ionizing radiation and displacement damage as a result of radiation

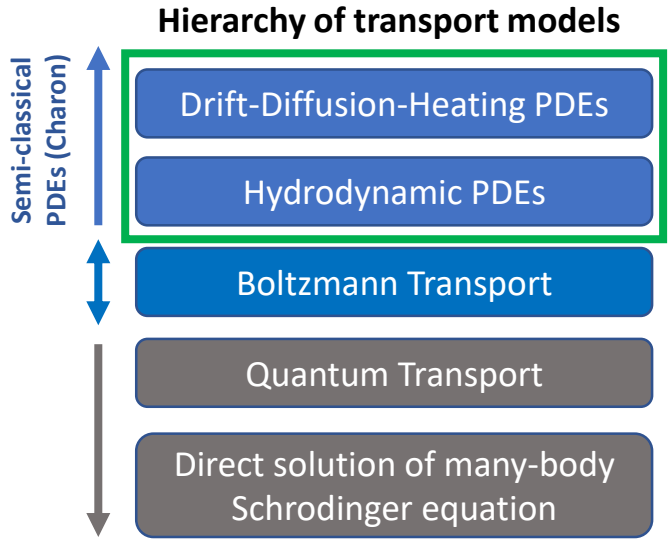
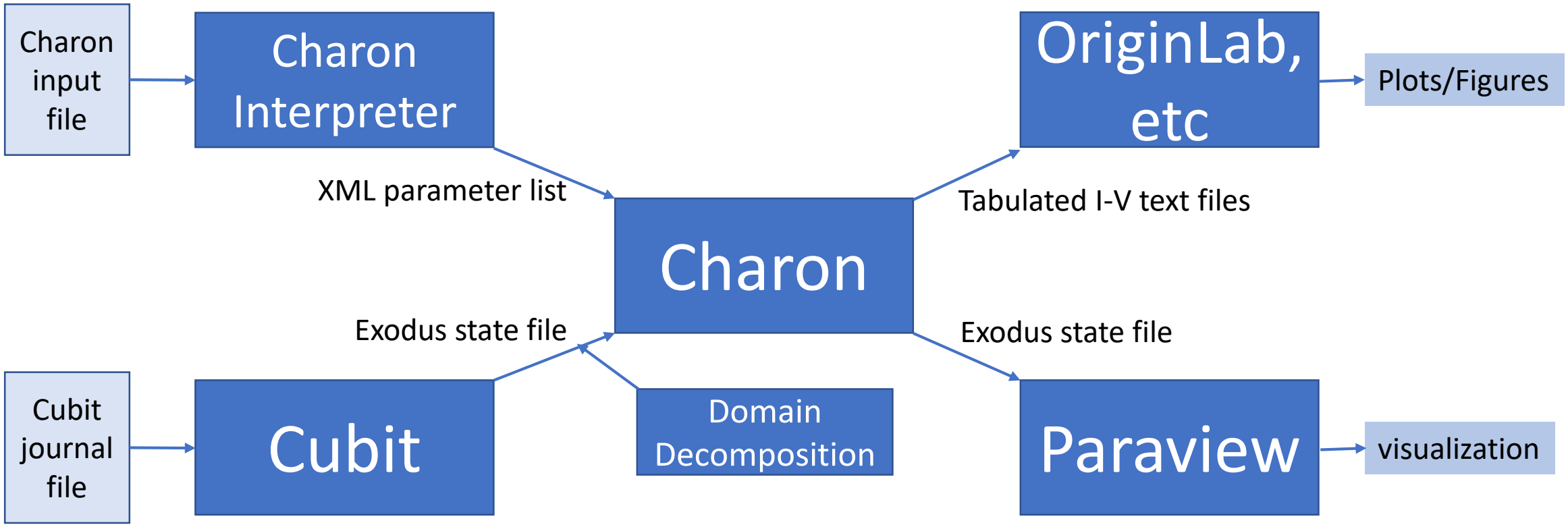


Diagram of a Simulation



Charon Files

- Input files
  - Contain state file names and a full parametric description of the simulation
- State files
  - Input
    - Meshing and initial guesses for a simulation
  - Output
    - Results of a simulation
- Various text files
  - Input
    - Assorted radiation data: pulse information, etc
  - Output
    - Tabulated I-V or I-<parameter> data from a transient or parameter sweep simulation

Charon Environment & Tutorials

- Copy the tutorial problems to your personal account
  - cp -R /tcad-charon/docs/Training/BasicIntroduction.
  - Two tutorial directories
  - Training slides (pdf)
- Cubit is usually the best option for mesh generation
  - Cubit is commercial, but a free limited version is available at:
    - https://www.coreform.com/products/coreform-cubit/free-meshing-software/

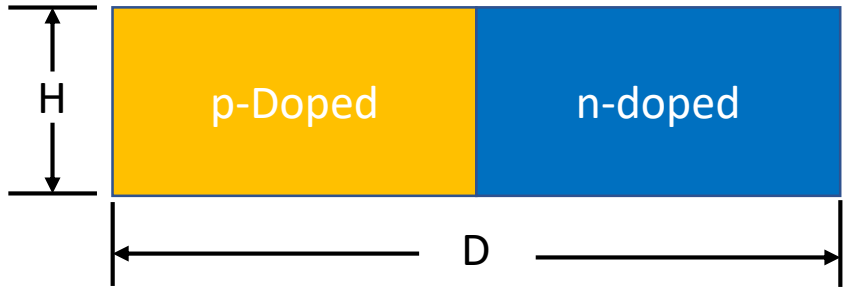
Charon Interpreter

- Charon Interpreter
  - Python driven front end to Charon
  - Simple, readable syntax
  - Straightforward formatting
  - Available syntax reference, searchable through “less”
  - Maps input syntax to Teuchos parameter list (xml) that fully configures a simulation
  - Can execute Charon in parallel or serial
  - (in the future) Will perform domain decomposition for parallel simulations

CharonInterpreter Help (exercise)

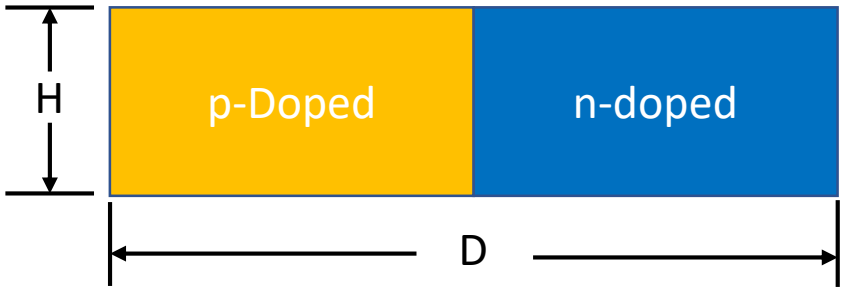
```
lcmusso@ceerws1810 [~] % charonInterpreter --help
usage: charonInterpreter [-h] [-i INPUT] [-r] [--no_current] [--np NP] [-s]
                         [-S] [-v VERBOSITY] [-p PATH_TO_CHARON]

optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        Specify the interpreter input file
  -r, --run             Execute Charon
  --no_current          Disable text output of current
  --np NP               Specify the number of processors to use in the Charon
                        run.
  -s, --syntax          Get help for syntax of interpreter. Append "| less -i"
                        to scrolling and search.
  -S, --longsyntax      Get expanded help for syntax of interpreter. Append "|
                        less -i" to scrolling and search.
  -v VERBOSITY, --verbosity VERBOSITY
                        Specify verbose output of the interpreter ranging from
                        0 to 25
  -p PATH_TO_CHARON, --path_to_charon PATH_TO_CHARON
                        Specify path to Charon exectutable
lcmusso@ceerws1810 [~] %
```

Exercise – Run a diode simulation



- In the tutorial #1 directory
  - charonInterpreter.py -i pndiode.iv.inp --run
  - Open paraview
    - Load pndiode.iv.exo
    - View potential, electron density, hole density
  - Plot the IV curve
    - Open the currents-loca.dat file in an editor and place a # on the first line in front of headers
    - gnuplot
      - gnuplot> set logscale y
      - gnuplot> plot ‘currents-loca.dat’

Diode exercise—What are the numbers?

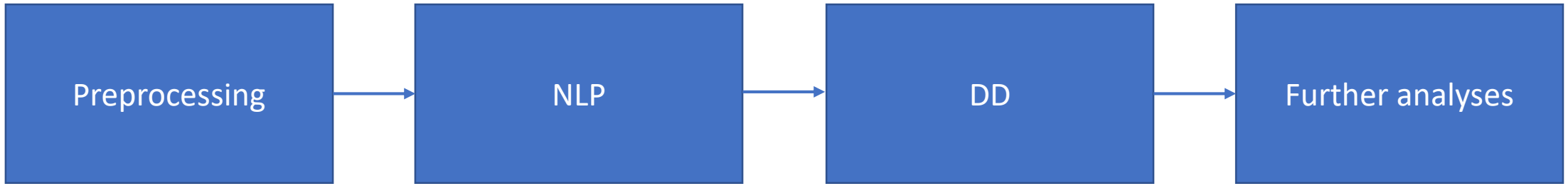


#Varying Voltage	cathode_si_Current	anode_si_Current
0.00000000e+00	1.37206471e-13	-6.85153401e-14
2.00000000e-02	1.93366012e-13	-1.89757579e-13
4.00000000e-02	5.99639521e-13	-5.96743300e-13
6.00000000e-02	1.47271575e-12	-1.47144327e-12
8.00000000e-02	3.34208701e-12	-3.34206878e-12
1.00000000e-01	7.34616330e-12	-7.34837395e-12
1.20000000e-01	1.59305344e-11	-1.59334701e-11
1.40000000e-01	3.43226611e-11	-3.43225631e-11
1.60000000e-01	7.37362115e-11	-7.37376517e-11
1.80000000e-01	1.58182023e-10	-1.58184588e-10
2.00000000e-01	3.39101247e-10	-3.39105026e-10
2.20000000e-01	7.26660146e-10	-7.26655744e-10
2.40000000e-01	1.55674147e-09	-1.55674237e-09
2.60000000e-01	3.33436167e-09	-3.33436276e-09
2.80000000e-01	7.14046286e-09	-7.14046237e-09
3.00000000e-01	1.52881663e-08	-1.52881650e-08
3.20000000e-01	3.27261345e-08	-3.27261311e-08
3.40000000e-01	7.00382567e-08	-7.00382547e-08
3.60000000e-01	1.49853191e-07	-1.49853192e-07
3.80000000e-01	3.20532857e-07	-3.20532857e-07
4.00000000e-01	6.85389084e-07	-6.85389082e-07
4.20000000e-01	1.46499725e-06	-1.46499725e-06
4.40000000e-01	3.12996977e-06	-3.12996977e-06
4.60000000e-01	6.68347406e-06	-6.68347406e-06

- Currents-loca.dat has I-V data
  - What are the currents?
  - Charon is 2D & 3D only
    - 2D: The current values will have units of Amps/cm width
    - 3D: The current values will have units of Amps
  - This simulation is set up as a quasi-1D problem
    - It is technically 2D, but all the action is in a single direction—H is an arbitrary length
    - Current is contactArea*current/H

Path to a Full Drift-Diffusion Simulation

- Two steps are required to produce a drift-diffusion (DD) simulation
- Usually, it isn't possible to go from a null state to a DD state in a single pass
- A nonlinear Poisson (NLP) solution is computed first for the electrostatic potential
- This is used as the initial guess for a DD solution



Cubit—Solid Model & Meshing

- Cubit is the Sandia tool for creating solid models and meshes
  - Not much covered in Charon basic introduction training
    - Introductory training taught by Cubit team should be sufficient
  - Most TCAD solid models are rectangular

Meshing the diode

Always use lower case for names

```
graphics mode wireframe

# Create a three-dimensional volume. The two-dimensional diode will be
# created on a surface of this three-dimensional volume. By default
# Charon assumes the dimensions are in microns
create brick x 1.0 y 0.5 z 0.1

# This makes the coordinates of the resulting mesh all positive. This
# isn't required but can be useful for post-processing
move vertex 4 location 0 0 0

# These will be the contacts, anode and cathode. The names are used in
# the input file to distinguish them
sideset 1 curve 3
sideset 1 name "anode"

sideset 2 curve 1
sideset 2 name "cathode"

# "blocks" are typically regions of different materials or distinct
# regions of the device. For this simple problem we only have one
# region
block 1 surface 1
block 1 name "si"

# Quads are currently the preferred element type for Charon
# simulations.
block 1 element type quad4


# The interval specifications set how dense or coarse the
# discretization is

## Long side
curve 2 4 interval 100

## Short side (contacts)
curve 3 1 interval 10

# Generate, or "mesh", the problem geometry
mesh surface 1

# Create the output "exodus" file, or overwrite it if it already
# exists.
export mesh "pndiode.exo" dimension 2 overwrite
```

Solid model (use microns)

Naming Contacts

Naming geometric regions

Meshing

Exporting mesh

Charon Input File

```
import state file pndiode.exo

start output parameters
  output state file pndiode.dd.iv.exo
end output parameters

start physics block Semiconductor
   geometry block is si
   standard discretization type is drift diffusion gfem
   material model is siliconParameter
end physics block

start material block siliconParameter
   material name is Silicon
   relative permittivity = 11.9

   start step junction doping
      acceptor concentration =1.0e16
      donor concentration =1.0e16
      junction location = 0.5
      dopant order is PN
      direction is x
   end step junction doping

end material block siliconParameter

initial conditions for ELECTRIC_POTENTIAL in si is exodus file
initial conditions for ELECTRON_DENSITY in si is equilibrium density
initial conditions for HOLE_DENSITY in si is equilibrium density

bc is ohmic for cathode on si fixed at 0.0
bc is ohmic for anode on si swept from 0.0 to 1.0

start solver block
  use solver pack 1
end solver block
```

- The highlights indicate where information supplied in the geometry/meshing phase of preprocessing ties into the Charon input
- The imported state file contains the mesh and a state—if there is one.
- The remainders are names tied to regions or boundaries of the device as named during the meshing phase.

charonInterpreter (chirp) Syntax Reference

- The interpreter can supply a syntax reference on command
  - charonInterpreter.py --syntax, or charonInterpreter -s
    - Provides an abbreviated help
  - charonInterpreter.py --longsyntax, or charonInterpreter -S
    - Provides a longer help
- Syntax help can be piped through less to make it scrollable and searchable
  - charonInterpreter.py -S | less -i
  - To search: /<search term>
- Exercise: Search the syntax help for “state file”

charonInterpreter Syntax Reference

{} indicates user-supplied entries—these are ALWAYS case sensitive

Import State File {filename} [at Index {index}]

[] indicates optional entries to the command

charonInterpreter Syntax Reference

BC is ohmic for {sidesetID} on {geometryBlock} [fixed at {potential} [swept from {potential1} to {potential2}]]

- Sometimes multiple options are available
  - Nested in the reference by [option1 [option2]]
  - Voltage on a contact can be fixed at a value
  - Voltage can be swept from one value to another
    - Such as an IV sweep
  - In this instance, at least one of the options must be selected

Input File Structure—Essential Elements

- Import/Export
  - Specifies state files to import for geometry, states
  - Specifies state files and other variables for export in the state file
- Physics Blocks
  - Sets the equations to be solved
  - Toggles various physics on & off
  - Ties physics to geometry and Material parameters in the Material Block
- Material Blocks
  - Defines material properties
- Initial Conditions
  - Specify what to use for initial values for variables (electrons, holes, potential, temperature)
- Boundary Conditions
  - Contact potentials, boundary temperatures
- Solver Specifications
  - Tolerances, preconditioner and solver methods

Input File Structure—State file import & Output

- Charon state files contain geometric information and solution
  - Exodus formatted
  - Geometric information includes the mesh
  - State can be null
    - File contains only a mesh
  - There can be multiple states
    - Output from transient or parameter sweep
- File names are specified in the input file for input and output state files

Input File Structure—State file import & Output

```
import state file pndiode.exo
```

- Imports the state file
  - Will contain at least the mesh and geometry
  - Might contain a state to use as an initial guess

```
start output parameters
   output state file pndiode.nlp.exo
end output parameters
```

- Specifies the output state file name
  - Will contain the mesh, geometry and the state just calculated
  - Can contain other directives for output of specific variables or tabulated data from sweeps (Exercise on this later)

Input File Structure—Physics Block

```
start physics block semiconductor
   geometry block is si
   standard discretization type is nlp
   material model is siliconParameter
end physics block semiconductor
```

- Physics block contains information about a region
  - The block must have its own unique name (semiconductor)
  - The corresponding region name from geometry
  - The equations to be solved
  - The name of the associated material block

Input File Structure—Material Block

```
start material block siliconParameter
   material name is Silicon
   relative permittivity = 11.9
   start step junction doping
      acceptor concentration = 1e16
      donor concentration = 1e16
      junction location = 0.5
      dopant order is PN
      direction is x
   end step junction doping
end material block siliconParameter
```

- Material block contains material property information
  - Must have its own unique name
  - Specifies a material name for parameters (Silicon)
  - Specifies the doping
  - Potentially numerous other material properties (mobility) if not defined internally or a different value from default is desired

Input File Structure—Boundary Conditions

```
bc is ohmic for anode on si fixed at 0
bc is ohmic for cathode on si fixed at 0
```

- Boundary conditions
  - Specifies the voltage on the contacts or heat flux/temperature at a boundary

Input File Structure—Initial Conditions

```
initial conditions for ELECTRIC_POTENTIAL in si is equilibrium potential
```

- Specifies initial conditions to the solve
  - Could be uninitialized
  - Could be a model—an estimation of the solution at equilibrium
  - Could be “exodus file” if read from a previous solution

Input File Structure—Solver Block

```
start solver block
   use solver pack 1
   nonlinear solver tolerance = 1e-6
end solver block
```

- Solver block specifies solver parameters
  - Specifies which solvers and preconditioners to use
  - Specifies tolerances for nonlinear solves

Exercise—Run a Diode Start to Finish

- From the Tutorial #2 directory, run the full sequence of tools to produce the same diode data as the first exercise
  - Generate the mesh
  - Decompose the mesh for a parallel run
  - Run the nonlinear Poisson solve
  - Run the drift-diffusion solve with anode sweep

Exercise—Generate and Decompose the Mesh

- cubit pndiode.jou
  - Generates the mesh
  - Use cubit to examine (it’s not exciting)

- decomp --processors 4 pndiode.exo
  - Decompose the mesh with the decomp tool

Exercise—Run the nonlinear Poisson Solve

- charonInterpreter.py -i pndiode.nlp.inp --np 4 --run
  - Use paraview to examine the potential field

Exercise—Run the Drift-Diffusion Solve

- charonInterpreter.py -i pndiode.iv.inp --np 4 --run
  - Note the initial conditions in the input file
  - Examine the potential and carrier densities over the sweep in paraview
  - Compare the iv data to the first diode exercise
  - Use paraview to examine the potential / depletion region

Exercise—Extra Credit Problem

- Use the interpreter reference to:
  - Name the file for iv output srh-currents.dat
    - Add to the output parameters block
  - Add srh recombination
    - Toggle on srh recombination in the physics block
    - Add srh lifetimes to the material block (fixed 1e-9 for both electrons and holes)
- Need only run the iv sweep
  - All else including nonlinear Poisson solve remains unchanged
- Compare the iv sweep with and without recombination
    - Open the currents-loca.dat file in an editor and place a # on the first line in front of headers
    - gnuplot
      - gnuplot> set logscale y
      - gnuplot> plot ‘currents-loca.dat’,’srh-currents.dat’

Exercise—
Extra Credit Problem

```
import state file pndiode.exo

start output parameters
  output state file pndiode.dd.iv.exo
  Output tabulated parameter currents to srh-currents.dat
end output parameters

start physics block Semiconductor
   geometry block is si
   standard discretization type is drift diffusion gfem
   material model is siliconParameter
   srh recombination is on
end physics block

start material block siliconParameter
   material name is Silicon
   relative permittivity = 11.9

   start step junction doping
      acceptor concentration =1.0e16
      donor concentration =1.0e16
      junction location = 0.5
      dopant order is PN
      direction is x
   end step junction doping

   start Carrier Lifetime block
      electron lifetime is constant = 1e-9
      hole lifetime is constant = 1e-9
   end Carrier Lifetime block

end material block siliconParameter

initial conditions for ELECTRIC_POTENTIAL in si is exodus file
initial conditions for ELECTRON_DENSITY in si is equilibrium density
initial conditions for HOLE_DENSITY in si is equilibrium density

start sweep options
   initial step size = 0.02
   minimum step size = 0.02
   maximum step size = 0.02
end sweep options

bc is ohmic for cathode on si fixed at 0.0
bc is ohmic for anode on si swept from 0.0 to 1.0

start solver block
  use solver pack 2
  nonlinear solver tolerance = 1e-6
end solver block
```