1 Doping Profile Mapping

Today's work focused on understanding how the doping profile maps onto the simulation grids in both Galene and our own simulator. This mapping is crucial for accurately comparing simulation results. Currently, we are using a uniform grid spacing of 1 nm. However, to precisely analyze the doping profile's mapping, we plan to double the grid density by halving the grid spacing.

Several essential plots remain pending, notably:

1. A current comparison plot between Galene and our simulator.
2. A detailed comparison plot after refining the grid.

These plots will provide deeper insights into discrepancies and alignments between both simulation approaches.

2 About the Codes

How it works internally:

1. **Direct Binary Reading:** The `gal_file_init` subroutine (lines 126–197) directly reads the binary OSV file using Fortran's unformatted file access. It doesn't need conversion to ASCII first.
2. **Block-based Structure:** The OSV file contains data blocks with a specific binary format:
   - Each block has a data type indicator (integer, real, or character)
   - Block header contains: number of data elements + size of name
   - Block name (character string)
   - Block data (based on type)
3. **Data Extraction:** The `gal_file_get_block` function (lines 199–212) retrieves specific data blocks by name, similar to what `gal3_ex` would do.

Key differences from command-line approach:
Instead of:

- `gal3 -c XXX.OSV` Convert to ASCII
- `gal3_ex -extract ...` Extract values

The code does:

- `call gal_fl%init("DD.OSV")` Direct binary read
- `b => gal_fl%get_block("n-density")` Extract by name
- `values = b%rdata` Get real data

How it processes the data:

1. **Units & Normalization:** Lines 134–168 define physical units for various quantities, and `normalization_m` handles unit conversions automatically.

2. **Grid Integration:** In `device_params.f90`, the galene data is integrated into the grid structure:

   - Line 178: `call this%gal_fl%init(gal_filename%s)` — loads the OSV file
   - Lines 184–194: Extract material information and properties
   - Lines 517–534: Use volume descriptions to set up the computational grid

3. **Data Types:** The code handles three data types (lines 17–19):

   - `GALDATA_INT = 1` (integer data)
   - `GALDATA_REAL = 2` (real data)
   - `GALDATA_CHAR = 3` (character data)

The m4 macros handle platform-specific details like integer sizes and ensure compatibility across different systems.

This approach eliminates the need for external preprocessing tools by implementing the OSV file reader directly in Fortran, providing seamless integration with the simulation workflow.