Great — this is step **4 of `parse_spd`**, where the parser takes the **IC/decap `.Connect` blocks** and actually turns them into **via lists with coordinates and types**. Let's turn `_fill_ic_decap_vias` into documentation.

📘 Documentation

`_fill_ic_decap_vias(brd, node_info, ic_blocks, decap_blocks, snap_dec=7)`

Purpose

Populate the PDN board object `brd` with the **IC and decap vias** defined in the SPD file.

Outputs are:

- `brd.ic_node_names` → list of IC node names in order
- `brd.ic_via_xy` → array of IC via coordinates (meters)
- `brd.ic_via_type` → array of IC via types (`0 = GND`, `1 = PWR`)
- `brd.decap_node_names` → list of decap node names in order
- `brd.decap_via_xy` → array of decap via coordinates (meters)
- `brd.decap_via_type` → array of decap via types (`0 = GND`, `1 = PWR`)

How it works

1. Iterate over `.Connect` blocks

Each IC/decap port from `_extract_connect_blocks` looks like:

```
bashCopy code1 $Package.Node10
2 $Package.Node12
```

- Lines beginning with **1** = "plus" side (usually **PWR** via).
- Lines beginning with **2** = "minus" side (usually **GND** via).

The function processes both.

---

2. Extract node names

```
pythonCopy codem = re.search(r"\$Package\.(Node\d+)", ln)
node = canon_node(m.group(1))
```

- Pulls out the `NodeXX` name from `$Package.NodeXX`.
- Canonicalizes it (`Node013 → Node13`).

---

3. Collect via info

For each node:

- Looks it up in `node_info` (built earlier from `_extract_nodes`).
- Appends:
  - **Name** → `names_ordered` list.
  - **Coordinates** → `[x, y]` converted from **mm → meters**.
  - **Type** → determined by `net`:
    - gnd → `0`
    - `pwr…` → 1
    - otherwise → fallback `0`.

---

4. Convert to arrays

```
pythonCopy codexy_arr = np.round(np.array(xy_list, dtype=float), snap_dec)
type_arr = np.array(type_list, dtype=int)
```

- Coordinates are rounded to `snap_dec` decimal places (default = 7).
- Returns numpy arrays for easier later computation.

If no vias exist in the block → returns empty arrays.

---

5. Assign to `brd`

```
pythonCopy codebrd.ic_node_names = ic_names
brd.ic_via_xy = ic_xy
brd.ic_via_type = ic_type
```

```
brd.decap_node_names = decap_names
brd.decap_via_xy = decap_xy
brd.decap_via_type = decap_type
```

Example

SPD excerpt:

```
bashCopy code.Connect ic_port1
1 $Package.Node12
2 $Package.Node10
.EndC

.Connect decap_port1
1 $Package.Node14
2 $Package.Node16
.EndC
```

From `node_info`:

```
bashCopy codeNode10 → {'net': 'gnd', 'x': 28.0, 'y': 30.0, 'layer': 1}
Node12 → {'net': 'pwr', 'x': 25.0, 'y': 25.0, 'layer': 3}
Node14 → {'net': 'gnd', 'x': 28.0, 'y': 25.0, 'layer': 4}
Node16 → {'net': 'pwr', 'x': 20.0, 'y': 20.0, 'layer': 5}
```

Parsed result:

```
pythonCopy codebrd.ic_node_names = ["Node12", "Node10"]
brd.ic_via_xy = np.array([[0.025, 0.025], [0.028, 0.030]])
brd.ic_via_type = np.array([1, 0])   # PWR, GND

brd.decap_node_names = ["Node14", "Node16"]
brd.decap_via_xy = np.array([[0.028, 0.025], [0.020, 0.020]])
brd.decap_via_type = np.array([0, 1])  # GND, PWR
```

Why it matters

- This step **links `.Connect` blocks to actual via coordinates** from `node_info`.
- Ensures correct port ordering: all **PWR vias first**, then **GND vias**, matching PowerSI conventions.

- Later, these arrays feed into the boundary integration and impedance calculation.

 Summary:
`_fill_ic_decap_vias` turns raw `.Connect` blocks into structured arrays of IC and decap via names, coordinates (meters), and types (PWR/GND). These become the core via definitions inside the `brd` PDN model.

Would you like me to continue this documentation trail with **`_extract_via_lines`** (the next parser stage) so you'll have the whole chain covered?