Absolutely Let's turn both `_read` and `_extract_board_polygons` into clear **developer-style documentation** that you can drop into your project. I'll explain purpose, inputs, outputs, and an example using your `b4_1.spd`.

Documentation

`_read(path: str) -> str`

Purpose

Reads a PowerSI `.spd` file from disk and returns its **entire content as a single string**.
It handles file validation, encoding issues, and provides clear error messages.

Behavior

1. **Validation**
    - Checks that `path` is a non-empty string.
    - Ensures the file exists.
2. **Read attempt**
    - Opens file in text mode (`utf-8`) and reads all lines into one string.
3. **Fallback**
    - If decoding fails (e.g. strange encoding), retries with `errors="ignore"`.
    - Raises explicit errors for missing files, permission issues, or unreadable binary content.

Inputs

- `path (str)`: Path to the `.spd` file.

Outputs

- `str`: Entire SPD file content as a string.

Example

```
pythonCopy codetext = _read("b4_1.spd")
print(len(text))  # total characters
print(text[:200]) # preview of first lines
```

`_extract_board_polygons(text: str, tol: float = 1e-9) -> List[np.ndarray]`

Purpose

Parses `.Shape` definitions in the SPD file to extract the **board outline polygon(s)**.
Returns each polygon as an `N×2` NumPy array of coordinates (in **meters**).

Behavior

1. **Find shapes**
    - Extracts `.Shape ShapeSignalXX …` blocks and their geometry:
        - `Polygon x1 y1 x2 y2 … mm` → arbitrary polygon.
        - `Box x0 y0 w h mm` → expanded into a rectangle.
2. **Map to layers**
    - Reads `PatchSignalXX Shape=ShapeSignalYY Layer=SignalZZ` lines.
    - Creates a mapping of which shape belongs to which layer.
    - Sorts by layer index (`Signal01`, `Signal02`, …).
3. **Parse coordinates**
    - Converts all `mm` values to floats.
    - Groups into `(x, y)` coordinate pairs.
    - Ensures polygons are closed (last point = first point).
    - Converts from **mm → meters** by multiplying with `1e-3`.
4. **Fallbacks**
    - If the strict regex misses shapes, uses a more permissive search.
    - Supports multiple `Box` lines in one shape.
5. **Deduplication**
    - `_collapse` checks if multiple shapes are identical (within `tol`).
    - If so, returns only one copy.

Inputs

- `text (str)`: Full SPD file content (from `_read`).
- `tol (float)`: Coordinate tolerance for considering shapes identical.

Outputs

- `List[np.ndarray]`: Each polygon as an `N×2` array in meters.
  (If no shapes found, returns `[]`.)

Example (from `b4_1.spd`)

SPD content snippet:

```
diffCopy code.Shape ShapeSignal7
Polygon 0.000000e+00mm 0.000000e+00mm
+ 4.000000e+01mm 0.000000e+00mm
+ 4.000000e+01mm 4.000000e+01mm
+ 2.000000e+01mm 4.000000e+01mm
+ 2.000000e+01mm 2.000000e+01mm
+ 0.000000e+00mm 2.000000e+01mm
+ 0.000000e+00mm 0.000000e+00mm
.EndShape

PatchSignal1 Shape = ShapeSignal7 Layer = Signal1
```

Parsed result:

```
pythonCopy code[
  array([
    [0.00, 0.00],
    [0.04, 0.00],
    [0.04, 0.04],
    [0.02, 0.04],
    [0.02, 0.02],
    [0.00, 0.02],
    [0.00, 0.00]
  ])
]
```

 Together:

- `_read` → gets raw SPD text.

- `_extract_board_polygons` → finds `.Shape` + `PatchSignal` lines, builds the **board outline polygon(s)**, and stores them in `brd.bxy`.

Do you want me to also write the documentation in **docstring format** (so you can paste directly into the code above each function), or do you prefer to keep it as this external Markdown/README-style explanation?