Please, Fold the Line: Designing Flexible Electronics Using Open-Source Software

Nico Arnold, Andreas Krinke, Manfred Dietrich, Jens Lienig
Institute of Electromechanical and Electronic Design (IFTE)
Dresden University of Technology, Dresden, Germany
Email: {nico.arnold, andreas.krinke, manfred.dietrich}@tu-dresden.de, jens@ieee.org

***Abstract*—Predicting the properties of flexible printed circuits (FPCs) before production is a significant challenge in their development. Traditional design flows model FPCs in a flat state, yet simulations of the actual bent shape of the circuits are more effective in forecasting their behavior in real-world applications. Currently, only proprietary tools are available for simulating these 3D geometries, with no open-source alternatives. This paper introduces a novel open-source software tool that bridges this gap by transforming two-dimensional layout data from KiCad files into three-dimensional bent geometries. This transformation is achieved through segmentation and meshing techniques, enabling accurate verification of circuit parameters via simulation. The software facilitates these simulations by generating the necessary input files, thus supporting a comprehensive open-source design and verification workflow. This work not only enhances the accessibility of FPC design and simulation but also underscores the broader applicability of open-source solutions for printed circuit board design.**

Keywords—flexible circuit, open source, simulation, FEM, FPC

I. Introduction

Flexible printed circuits (FPCs) provide significant advantages over traditional rigid printed circuit boards (PCBs). Particularly beneficial in projects with complex geometries or limited space, FPCs utilize a bendable substrate (often polyimide (PI)), in contrast to the fiberglass-reinforced epoxy (FR4) [1] used in rigid PCBs. The inherent flexibility of FPCs enables circuits that can adapt to curved surfaces or be folded to reduce their spatial footprint. Additionally, the material properties of PI are advantageous in high-frequency (HF) applications [2]. Despite these benefits, designing FPCs poses substantial challenges. Conventional layout designs are executed in 2D, yet the final form of these circuits is inherently three-dimensional, introducing complexities in visualizing and planning the final product. Effective design processes must preemptively address potential issues, such as component interference, and unwanted electrical or thermal interactions, such as crosstalk. By ensuring precise placement of components in the design phase, it is possible to facilitate assembly without unforeseen violations, and proper functionality in the finished 3D structure.

To date, the respective software is only available under proprietary licenses from companies like Altium and Cadence. For smaller companies and projects, the licensing cost is often too high, raising the demand for a free or open-source alternative. While existing open-source PCB design tools like KiCad [3] are widely used to design rigid PCBs, they lack capabilities to visualize or simulate FPCs in their bent configurations. Although simulations of rigid PCBs and flat FPCs can be performed with available open-source tools [4], [5], there are no known open-source tools for visualizing or simulating FPCs in their final, three-dimensional form.

In this paper, we present a workflow that exclusively utilizes open-source software to visualize and simulate flexible circuits. We have developed an application that allows users to bend circuits designed in the open-source program KiCad by applying specific bending parameters, filling the gap between FPC design and simulation. To illustrate the capabilities of this workflow, we use the design of the example circuit in Section III to demonstrate the transformation to its bent configuration and the thermal simulation of the generated geometry.

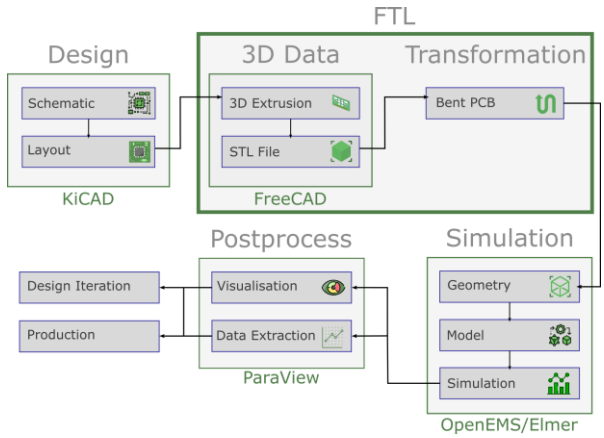

Figure 1 Common simulation workflow with our contribution to transform geometries highlighted

The source code of our program is licensed as open-source software and available on GitHub [6] for anyone to use and modify.

II. Method

In this section, we first describe our workflow for designing an FPC as shown in Fig. 1 (Sec. II-A), and then explain the detailed implementation of our new program (Sec. II-B).

A. Workflow

Our workflow begins with the design of the circuit as a flat PCB in KiCad [3]. Extra space needs to be reserved for the bending areas, as visible in the middle of the PCB shown in Fig. 2. A KiCad plugin was created to enable users to draw the bending areas and adjust their parameters (e.g. bending angle,

Please cite as: N. Arnold, A. Krinke, M. Dietrich, J. Lienig "Please, Fold the Line: Designing Flexible Electronics Using Open-Source Software," Proc. 6th IEEE Int. Flexible Electronics Technology Conf. (IFETC'24), Bologna, Italy, Sept. 2024.

direction). This information is then exported as a JSON file together with the paths to the 3D (STL) files of the individual layers.

The KiCad layout file (.kicad_pcb) is used to generate the STL files of the substrate and the copper traces by using the macro fcad_pcb [7] in FreeCAD [8]. This macro extrudes the 2D polygons representing the copper routes into a 3D stack of copper wires and substrate layers. Starting from there, the 3D geometries are meshed and then exported as STL files.

We developed an open-source program to bend the 3D circuit geometry. The 3D meshes generated from kicad_pcb are used as a data source for the transformation algorithm explained in Sec. II-B. Before the transformation, the area to be bent is highlighted, as shown in Fig. 2. After confirmation, the algorithm is executed, the bent circuit board is visualized and can be exported to FreeCAD in order to populate parts.

The output of our program can be used for simulations. For this, the bent geometries can be further processed in SALOME [9] for advanced mesh operations and, for example, imported into Elmer [10] to enable Finite Element Method (FEM) simulations of the transformed circuit. Another option is to use straight geometries in OpenEMS [11] to execute Finite Difference Time Domain (FDTD) simulations, as the FDTD method is not suited for bent geometries.

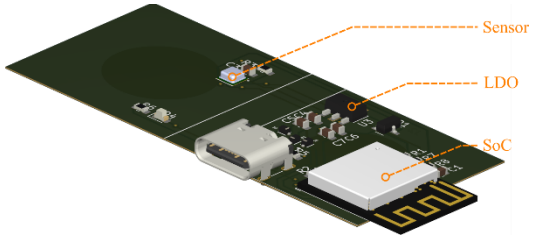

Figure 2 FPC rendered in KiCad's 3D viewer

$$t = \frac{p_x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

$$\alpha = \alpha_{max} \cdot t \quad (2)$$

$$r = \frac{x_{max} - x_{min}}{\alpha_{max}} \quad (3)$$

$$\overrightarrow{p'} = \begin{bmatrix} p_x' \\ p_y' \\ p_z' \end{bmatrix} = T \cdot \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \quad (4)$$

$$T = \begin{bmatrix} 0 & 0 & -\sin(\alpha) & x_{min} + r\sin(\alpha) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos(\alpha) & r(1 - \cos(\alpha)) \end{bmatrix} \quad (5)$$

$$R = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) & -x_{max}\cos(\alpha) + x_{min} + r\sin(\alpha) \\ 0 & 1 & 0 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) & -x_{max}\sin(\alpha) + r(1 - \cos(\alpha)) \end{bmatrix} \quad (6)$$

B. Implementation

In our Python program, the first step is to parse and import the meshing information for copper and substrate layers from the JSON file. Then, the transformations are processed—every transformation includes a set of attributes to specify the designated area on the FPC, how many degrees the circuit is bent and in which direction. Transformations are hereby characterized as rectangles with coordinates x_{min}, x_{max}, y_{min} and y_{max}. Following that, the circuit geometry is sliced into sections according to those transformation areas. Beginning at the start line (red line in Fig. 2) with a bending angle of 0°, the algorithm then iterates over all the nodes of the mesh between there and the stop line of the specific transformation area. For every node with position $\vec{p} = [p_x, p_y, p_z]^T$, the position factor t on the transformation length is calculated (Eq. 1). Based on the maximum angle α_{max} configured in the JSON file and the value of t, the bending angle α is calculated for $\vec{p}$ so that the last points on the stop line are transformed with α_{max} (Eq. 2). Additionally, the bending radius r is determined by the width of the transformation rectangle (Eq. 3). For each node, the bend is then carried out using a coordinate transformation matrix T to calculate the new position $\vec{p}'$ of the node (Eq. 4). Eq. 5 shows an example transformation matrix T for a bend in positive z direction while moving in positive x direction relative to the flat PCB. If there are still geometries behind the stop line, those are rotated like the last points on the stop line so they form a straight plane tangentially to the transformed area. This is called a residual transformation based on a transformation matrix R as calculated in Eq. 6. The transformed points are then calculated just as in Eq. 4.

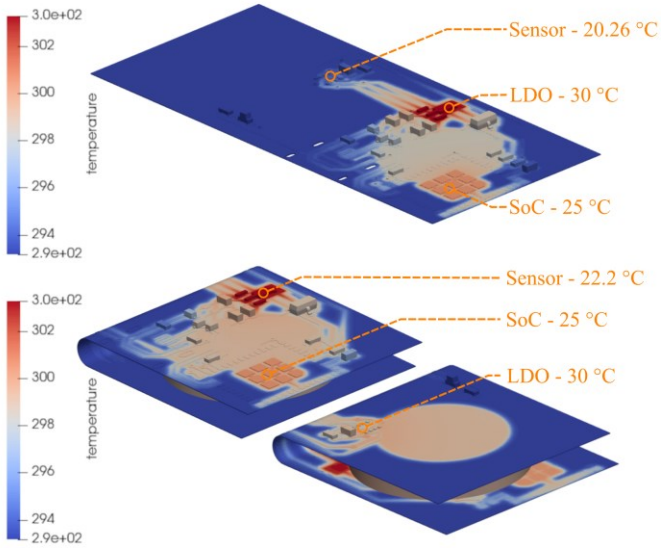

Figure 3 Thermal FEM simulation – flat board (top) vs. bent board (bottom)

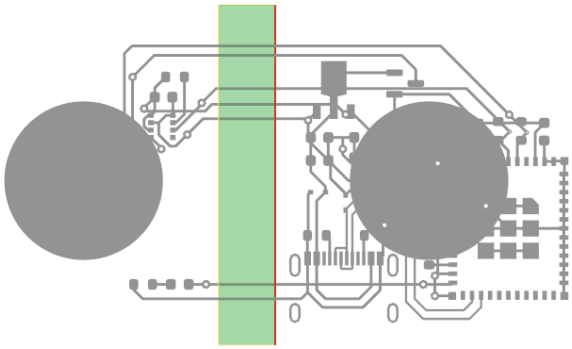

Figure 4 Bending area (green) and start line (red) as visualized in the application

III. Demonstrator and Results

We created a small test board containing an ESP32-H2 System-on-a-Chip (SoC) with several peripheral components to demonstrate the workflow. These include a low dropout (LDO) voltage converter for supplying the processor with power via USB, a BME280 temperature/humidity sensor and a CR2032 lithium coin cell as primary power supply. By bending the circuit board around the battery as illustrated in Fig. 2, there is no need for a separate battery holder. The SoC is able to control the sensor via I²C, read measurements, and transmit them through a ZigBee network.

During the operation of the processor and the voltage converter, heat is generated that could cause errors in the BME280's measurement results. Therefore, the temperature rise caused by heat dissipation is of interest. In the flat form (Fig. 3), the only way for the heat to reach the sensor is via the copper traces. As a significant amount of heat will be conducted by the battery when the FPC is bent around it, it is best to do a simulation, as the temperature at the sensor is expected to increase.

The processing of the bent geometry was then done in SALOME [9], which involves some manual adjustments. After defining physical groups, which later represent boundaries in the FEM simulation, the final mesh is generated with gmsh [12].

For this demonstration, the FEM simulation was set up in Elmer [10] with the pads under the LDO regulator and the SoC acting as heat sources with a constant temperature of 30 °C and 25 °C, respectively. The substrate was configured to emit heat with a thermal coefficient of 100 Wm^{-2} K^{-1} to the surrounding air with a temperature of 20 °C. The .vtu file generated by the Elmer solver is then fed into ParaView [13] to plot the temperatures on the mesh. It turned out that the temperature at the BME280 sensor was 20.26 °C in the flat configuration (Fig. 4 top). In the bent configuration with the battery as an additional thermal bridge (see Fig. 4 bottom), the temperature increased by nearly 2 K to 22.2 °C. The circular pad area heated by the battery (Fig. 4 left) can clearly be identified as the cause for the additional heating of the sensor.

IV. Conclusion

In this paper, we presented a comprehensive workflow for designing flexible printed circuits (FPCs) using exclusively open-source tools. Our tool transforms the 2D layout of a circuit board into a detailed flat 3D model that can then be bent. The result is visualized, creating a precise representation that allows for accurate measurement of dimensions, radii, and distances between points.

By rendering these models in 3D, users gain a clearer insight into the complex spatial configuration of flexible circuits, bridging the gap to mechanical CAD programs for further development of casings or mechanical components. The generated geometries facilitate the creation of accurate models for simulating crucial electrical, thermal, and mechanical parameters. This advancement empowers small companies and open-source projects to design, visualize, and simulate complex flexible circuits, thereby enhancing design accuracy and fostering innovation in FPC development.

References

[1] C. Fu, R. Brown, and C. Ume, "Temperature-dependent material characterizations for thin epoxy FR-4/E-Glass woven laminate," in Proc. of IEEE 43rd Electronic Components and Technology Conference (ECTC '93), 1993, pp. 560–562. [Online]. Available: https://doi.org/10.1109/ECTC.1993.346789

[2] M. Wagih, Y. Wei, and S. Beeby, "Flexible 2.4 GHz Node for Body Area Networks With a Compact High-Gain Planar Antenna," IEEE Antennas and Wireless Propagation Letters, vol. 18, no. 1, pp. 49–53, 2019. [Online]. Available: https://doi.org/10.1109/LAWP.2018.2880490

[3] "KiCad," 2024. [Online]. Available: https://www.kicad.org/

[4] S. Fießer and U. Schwalbe, "Analyzing and optimizing the switching behavior of power electronics by automated pcb parasitics extraction of the critical path," in Proc. of 23rd European Conference on Power Electronics and Applications (EPE'21 ECCE Europe), 2021, pp. P.1–P.8. [Online].
Available:
https://doi.org/10.23919/EPE21ECCEEurope50061.2021.9570455

[5] R. Szewczyk, A. Ostaszewska-Liz˙ewska, and P. Ra˚back, "Modelling the Fluxgate Sensors with Magnetic Field Concentrators," Acta Physica Polonica A, vol. 137, pp. 700–702, May 2020. [Online]. Available: https://doi.org/10.12693/APhysPolA.137.700

[6] N. Arnold, "IFTE-EDA/FTL: Fold the line – FPC geometry transformation and data extraction for visualization & simulation." [Online]. Available: https://github.com/IFTE-EDA/ftl

[7] Z. Lei, "Realthunder/FCAD_PCB: Freecad scripts for PCB CAD/CAM," 2024. [Online]. Available: https://github.com/realthunder/fcad_pcb

[8] "Your own 3d parametric modeler," 2024. [Online]. Available: https://www.freecad.org/?lang=en

[9] "SALOME Platform," 2024. [Online]. Available: https://www. salome-platform.org/

[10] "Elmer FEM." [Online]. Available: https://www.elmerfem.org/

[11] T. Liebig. openEMS - Open Electromagnetic Field Solver. General and Theoretical Electrical Engineering (ATE), University of Duisburg-Essen. [Online]. Available: https://www.openEMS.de

[12] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities," International Journal for Numerical Methods in Engineering, vol. 79, pp. 1309–1331, Sep. 2009. [Online]. Available: https://doi.org/10.1002/nme.2579

[13] Kitware, "ParaView," 2024. [Online]. Available: https://www.paraview.org/