OpenMFDA: Microfluidic Design Automation in Three Dimensions

Ashton Snelgrove, Daniel Wakeham, Skylar Stockham, Scott R. Temple, Pierre-Emmanuel Gaillardon
Dept. of Electrical and Computer Engineering
University of Utah
Salt Lake City, Utah, USA
pierre-emmanuel.gaillardon@utah.edu

***Abstract*—Current microfluidic design automation (MFDA) solutions are limited by the planarity requirements of current manufacturing techniques. Recent advances in stereolithography 3D printing create an opportunity for new MFDA design methodologies. We propose a methodology for the placement of microfluidic components and the routing of flow and control channels in three dimensions. Additionally, we propose a methodology for generating a printable 3D structure from the layout. We then present OpenMFDA, an open-source MFDA design flow implementing the proposed methodologies. This design flow takes a structural netlist and produces a sliced design for manufacturing using an SLA 3D printer. Our methodology demonstrates short run times and generates devices with 2-20× smaller area compared to state-of-the-art MFDA tools.**

***Index Terms*—Microfluidics, Electronic Design Automation, 3D Printing, Placement and Routing**

I. Introduction

Microfluidics, characterized by the precise manipulation of fluids at the micrometer scale, has emerged as a transformative technology, with the potential to revolutionize various scientific and industrial domains. Its applications span from analytical chemistry and biochemistry to medical diagnostics and drug delivery systems. Microfluidic "lab-on-a-chip" devices offer complex operations in circuits the size of a coin.

The field of microfluidic design automation (MFDA) is still in its early stages compared with the maturity of electronic design automation (EDA). Practitioners currently use manual processes in general purpose computer-aided design (CAD) tools to design their devices, with little use of automation [1]. While EDA tools are capable of generating devices with billions of transistors, MFDA tools are limited to dozens of components.

The current state-of-the-art manufacturing technique uses polydimethylsiloxane (PDMS) [2]. Chips consist of a single flow layer and a single control layer. This places a requirement of planarity, with a channel unable to cross another channel on the same layer [3]. This limitation greatly reduces placement and routing (PnR) options, creating a more complex problem formulation to solve with less available routing area. In contrast, semiconductor manufacturing processes on silicon typically provide multiple metal layers for signal routing, with state-of-the-art technology nodes offering a dozen metal layers or more. Recent innovations in stereolithography (SLA) 3D printing [4] create the opportunity to reformulate the traditional MFDA PnR problem. Channels and components are no longer limited to the planarity requirements of two fluid channel layers, creating additional flexibility in PnR.

The main contributions of this work are:

1) a methodology for placement of microfluidic components, and routing of flow and control channels in three dimensions,
2) a tool for generating a printable 3D model from the generated layout,
3) an MFDA design flow, *OpenMFDA*, which takes a structural netlist and generates manufacturing instructions for a 3D resin printer.

Our design flow is capable of generating a manufacturable device design in less than 15 seconds, comparable with the fastest state-of-the-art MFDA tool Columba [3] and orders of magnitude faster than the other state-of-the-art MFDA tool Fluigi [5]. The resulting design has 2-20× improvement in effective area while maintaining short channel lengths.

The rest of the paper is organized as follows. Section II explains the current state of microfluidic design and existing MFDA tools. Section III proposes our MFDA methodologies and the architecture of the MFDA workflow. Section IVintroduces the implementation of the methodology. V presents results and performance characteristics of benchmark devices. Section VI discusses the outcomes of this work and future paths forward. Section VII contextualizes results and concludes the paper.

II. Background

A. Overview of microfluidics

The field of microfluidics is concerned with the manipulation and analysis of fluids at the micrometer scale. Microfluidics, at its core, focuses on the precise control and manipulation of small volumes of liquids within microchannels, typically at the scale of microliters or nanoliters. This technology has found diverse applications across various domains, including biology, chemistry, medicine, and environmental science [6].

This material is based upon work supported by the National Science Foundation under Grant No. 2140148 and Grant No. 2245494. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

978-3-9826741-0-0/DATE25/© 2025 EDAA

B. EDA and MFDA parallels

The design of microfluidic devices draws parallels to the field of electronics. Much like electronic integrated circuits, microfluidic devices aim to miniaturize and integrate multiple functions onto a single chip, enabling precise control, mixing, and analysis of fluids in a defined and repeatable process. At the scale of microfluidics, fluids stay in laminar flow, and gravity is not a consequential factor. This simplifies the modeling of fluid behavior. Properties like pressure, cross-section, and channel length have analogs with electrical phenomena like voltage, current, and resistance. Valves and other active components have analogs with transistors. These analogies have been leveraged to apply existing EDA tools to problems in MFDA. Notably, fluidic simulation has been done using electronic simulation tools like SPICE [7][8].

C. State-of-the-art MFDA algorithms

In past literature, MFDA algorithms have focused primarily on the considerations of performance and cost-minimization, control-port minimization, meeting synchronization or timing, and reliability requirements. Performance and cost optimization includes techniques like clustering and partitioning, allowing denser placement of components [9]. Control-port minimization algorithms optimize the placement and routing of control signals to maximize the amount of control signals possible with minimal control inputs. This is especially relevant in the MFDA domain because physical space for control signals ports is very limited [10]. Synchronization-oriented algorithms ensure consistent fluid signal arrival times which is important in chip applications requiring precise fluid signal coordination for precise mixing and dosing [11]. Reliability-oriented considerations address fault tolerance, redundancy, and the mitigation of manufacturing variations, contributing to the overall robustness of the designed chips [12].

D. State-of-the-art MFDA design flows

Two state-of-the-art MFDA design flows are currently available, Columba [13] and Fluigi [5]. Features of each tool and our proposed design flow are compared in Table I.

1) Columba: Columba provides a design flow targeting PDMS chips [13]. Columba provides only two components in its module library, a multi-purpose ring mixer and a mixing chamber, with no capability of adding new components. Designs are represented in text as a netlist using a domain specific language (DSL) [14]. The placement and routing algorithm guarantees planarity for both control and flow layers. Control pin utilization is optimized, with a multiplexing scheme used to reduce the number of required control ports. Columba generates AutoCAD-compatible files that can be used for mask fabrication. The current version offers a hosted web interface [3]. Source code is not publicly available.

2) Fluigi and 3DμF: Fluigi also provides a design flow targeting PDMS chips [5]. The module library contains a larger set of components, including serpentine channels, cell traps, multiplexers, and droplet generation. Designs can be represented in text using a DSL, or a companion tool called 3DμF [15] which presents a graphical user interface for laying out components and channels. Output to a manufacturing protocol MakerFluidics is supported as well as generation of photo-masks. The tool can be run as a cloud hosted application [5]. Source code is publicly available under an open-source license.

E. 3D printing developments for microfluidics

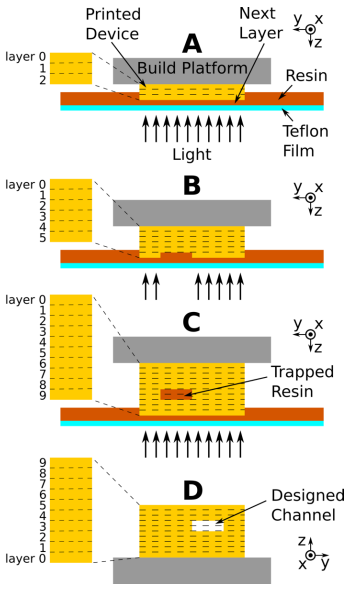

Fig. 1. Stereolithography (SLA) 3D printer operation [4]. (a) The build platform is submerged in photo-sensitive resin. The resin is exposed to a pattern of ultraviolet light. (b) The cured layer is pulled up out of the resin and a new layer is formed. (c) Negative space is created by selectively curing areas. Uncured resin remains trapped in features and must be flushed. (d) The result is a cavity for fluids.

State-of-the-art manufacturing currently uses injection molded polymers, with the most widely used being PDMS [2]. PDMS manufacturing usually requires the creation of a master mold for injection molding [16]. Injection molding offers very high throughput and detail, but very high setup costs for creating molds and the injection machinery. Other techniques are possible, including soft-lithography and plastic micro-milling [8][17].

PDMS devices are created in multiple layers, which must then be fused together over a membrane to create valves. A vertical integration using this method becomes infeasible due to design and manufacturing complications. A larger number of layers increases manufacturing error due to possible misalignment. This limits most device designs to two layers over a single membrane, creating a single control and a single flow layer.

Previously, 3D printing in microfluidics has been demonstrated with fused deposition modeling (FDM), which deposits layers of melted polymer through an extrusion head. The disadvantages of FDM are printing time and feature size [18].

SLA 3D printing instead uses photo-reactive polymer resin, with operations shown in Figure 1. A layer of resin is selectively cured by exposure to ultraviolet light. Selectivity

Tool	Component library	Input format	Control synthesis	Dimensionality	Simulation	Manufacturing	License
OpenMFDA	yes	Verilog	no	3D	SPICE	Sliced 3D printing	MIT
Columba	no	DSL	yes	planar	no	AutoCAD for photo mask	Proprietary
Fluigi	yes	DSL (MINT)	no	planar	yes	photomasks	BSD-3

TABLE I
COMPARISON OF STATE-OF-THE-ART MFDA WORKFLOWS.

is done by masking, or by targeted light sources [19]. The printed structure is then lifted up and another layer is created. Resolutions of 7.6μm per pixel have been demonstrated, with further improvement predicted in the near future [20][4]. SLA printing provides high throughput, with print time defined by the number of layers (i.e. overall depth) of the design. Material costs are very low. Print time for a device of 19.35mm × 12.10mm × 80mm dimensions is 30 minutes, a function of the 80mm print depth [4]. Typical devices can be printed in 5-20 minutes.

III. METHODOLOGY

A. Placement and routing methodology

In our proposed methodology, we formulate the placement of microfluidic components and routing of channels as analogous to the placement of standard cells and routing of metal layers in conventional EDA backend flows. We demonstrate that a microfluidic design can be encoded into a format that can be solved using existing EDA algorithms and tools.

A fluidic device is composed of a set of components C and a set of channels E that form a directed hypergraph, with nodes C and edges E. Primary inputs and outputs are represented as special nodes. There is no distinction made between control and flow channels. Valves are not created as crossings of flow and control channels, but instead are specific library cells.

1) Placement: The problem is first specified in 3D Euclidean space. Width, height, and depth refer to length along the $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$ axes respectively. SLA printers have a fixed constrained print area, a rectangular area B on the $\hat{\mathbf{x}}\hat{\mathbf{y}}$ plane, with sides parallel to the $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ unit vectors. The bottom left corner of the bounding box is placed at the origin. All coordinates are therefore positive.

Each component $c_j \in C$ has an associated 3D volume V_j of arbitrary shape representing the physical shape, and a set of associated 3D volumes P_j representing the connections to channels. The volume V_j has a fixed position on $\hat{\mathbf{z}}$. For volume V_j, project the volume as an area on the $\hat{\mathbf{x}}\hat{\mathbf{y}}$ plane. The bounding box b_j is the minimum-sized rectangle that encloses the area of the projection (eq. 1). The origin O_j of the component is the bottom left corner of the bounding box.

$$b_j = \text{bound}(V_j \mapsto \hat{\mathbf{x}}\hat{\mathbf{y}}) \quad \forall j : c_j \in C \tag{1}$$

The placement problem can be specified as follows: for all components, there exists an assignment to the component origin (eq. 2) that satisfies the conditions: all component bounding boxes are contained in the device bounding box (eq. 3), and no component intersects with another (eq. 4).

$$\forall j : c_j \in C \quad \exists\ O_j = [x, y] \tag{2}$$

$$\forall j : c_j \in C \quad b_j \subseteq B \tag{3}$$

$$\forall j, k : c_j, c_k \in C \quad b_j \cap b_k = \emptyset \tag{4}$$

2) Routing: Each channel $e_i \in E$ has a design-specified cross section w_i, d_i, which is constant for the length of the channel. Let t_w, t_d be the minimum depth and width respectively between channels specified by design rules.

Define a set of equidistantly spaced planes $a_0, ..., a_n$ on $\hat{\mathbf{x}}\hat{\mathbf{y}}$. The distance β between plane a_n and a_{n+1} on $\hat{\mathbf{z}}$ is equal to the maximum cross section depth d_i of any channel plus t_d.

$$a_0, ..., a_n \in \hat{\mathbf{x}}\hat{\mathbf{y}} \tag{5}$$

$$\beta = \|a_{k+1}\hat{\mathbf{z}}\| - \|a_k\hat{\mathbf{z}}\| = \max_{i:e_i \in E} d_i + t_d \tag{6}$$

Any channel extruded along the $\hat{\mathbf{x}}\hat{\mathbf{y}}$ plane can therefore be contained between any planes a_k, a_{k+1}, and every channel extrusion meets the design rule t_d.

Let S be the set of slices $[s_0, ..., s_n]$, with s_k equal to the volume bounded between planes a_k and a_{k+1}.

$$S = [s_0, ..., s_n]; a_k \leq s_k < a_{k+1} \tag{7}$$

$$\upsilon_{k,i} = s_k \cap V_i \quad \forall i : c_i \in C, k : s_k \in S \tag{8}$$

$$o_{k,i} = \text{bound}(\upsilon_{k,i} \mapsto \hat{\mathbf{x}}\hat{\mathbf{y}}) \tag{9}$$

For all intersections of slices and component volumes $\upsilon_{k,i}$, the obstruction $o_{k,i}$ is the minimum rectangle that fully encloses $\upsilon_{k,i}$ projected onto $\hat{\mathbf{x}}\hat{\mathbf{y}}$. For all intersections of slices and component pins, a similar projection p_k is made.

The minimum bound for number of slices is some multiple of β such that all components are contained in the volume.

$$n\exists \quad \beta * n > \max_{i:c_i \in C} \|Vi\hat{\mathbf{z}}\| \tag{10}$$

Let L be an ordered set $[l_0, ..., l_n]$ of 2D Euclidean spaces. Each space has an associated slice s_n, and all obstructions $o_{m,j}\ \forall\ m : l_m \in L,\ j : c_j \in C$ are located in the space.

A segment is a line defined by two points on the plane, and the length is the distance between the two points. A route $R_{f,g}$ consists of a set of segments on plane l_f. Segments are aligned parallel with either the $\hat{\mathbf{x}}$ or $\hat{\mathbf{y}}$ axis. Segments are connected if the end point of one segment coincides with the line defined by the other segment. All connected segments have the same width. The length of a route is defined as the sum of the length of all segments in the route (eq. 11). Segments cannot

overlap other segments or obstructions $o_{k,i}$ on the same layer. Segments must have minimum distance from other segments to meet design rules.

$$\|R_{f,g}\| = \sum_{k} \|s_k\| \in R_{f,g} \tag{11}$$

A via is a point on the plane $l_k[x, y]$ with a corresponding point on $l_{k+1}[x, y]$. A via is connected to a segment on the same plane if the via point is coincident with the segment line. A route $R_{k,g}$ is connected to route $R_{k+1,h}$ if both routes are connected to a via. A pin is connected to a route if a segment in the route overlaps the projected pin area. The set of routes $R \in L$ and connected pins form a hypergraph.

Given the aforementioned constraints, the routing problem is to find a set of layers L with segments S that form a set of routes R, such that the graph formed by R is isomorphic to hypergraph edges E, while minimizing the size of L and total length of R.

$$E \cong R \in L, \quad \min(\|L\|), \quad \min(\|R\|) \forall R \in L \tag{12}$$

B. Encoding

The cell bounding box, pin areas, and the layer obstruction set are written as a MACRO cell in the standard library exchange format (LEF). Each layer is specified as a ROUTING layer and a via CUT layer. Once encoded, a solution can be found using existing EDA tools. For our implementation of the methodology, we used the OpenROAD PnR tool [21]. Additional implementation details are discussed in section IV.

C. 3D model generation methodology

The output of the PnR methodology is the layout, which consists of a set of origin points for components, a set of layers with line segments, and a set of vias. For each cell instance, component geometries are created and then translated on $\hat{\mathbf{x}}\hat{\mathbf{y}}$ to the point defined in the layout. The routing layers are then generated in the order specified in the layout. At slice s_k, for each channel segment in routing layer l_k, a rectangular cuboid is extruded between the points specified in the segment definition. The width and depth of the channel are specified by the design netlist. All channels are placed with the bottom face on the plane of the bottom face of slice a_k. By (eq. 6) the depth β is of a size sufficient that no channel violates design rule t_d.

For each via, a rectangular cuboid is extruded along the $\hat{\mathbf{z}}$ axis from the terminal segment point in the previous slice to the segment point in the next slice. This channel shares the same cross section dimensions as the channels it connects.

Once all slices have been generated, the chip interconnect is placed. The interconnect is a multi-port matrix of channel pins, which each pin having a micro-gasket to seal with the interconnect of another device or control system [22]. Unlike other structures, the interconnect is represented in positive space.

IV. Methodology implementation - OpenMFDA

We now present OpenMFDA, which implements our proposed methodologies in an automated design flow. The components of the design flow are shown in Figure 2.

1) Component library: Components are defined as standard cells with representations across multiple domains. Each cell has the following:

- A specified width and height, obstructions, and a set of pins. Pins have a location, width and height, and an assigned layer. This is the required information formulated in our methodology.
- An analog physical model written in Verilog-AMS for use with physical simulation.
- A 3D geometric representation for generating the printable device. This is described using the OpenSCAD 3D modeling language [24].

2) Hardware description: The device design is specified as a structural netlist in Verilog. Components are specified as instances of modules from the component library, and connections between components are specified as nets. This provides hierarchical organization, simple connections, and component reuse.

3) Simulation: The device netlist is converted to an electrical analog and simulated using the Xyce SPICE simulator [25]. The results show the flow rates in various channels over time and additionally chemical concentrations at different nodes in the circuit.

4) Layout generation: Components from the library and the layer configuration are encoded into a gate level analog using our proposed methodology. The result is stored in the EDA standard LEF technology file format. The device netlist and the LEF file are run with the OpenROAD PnR tool [21], which generates a layout in the standard design exchange format (DEF).

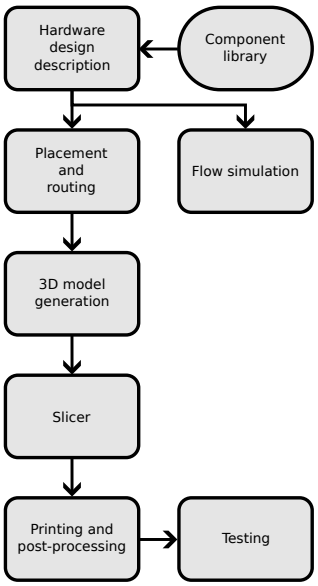

Fig. 2. OpenMFDA workflow diagram

Benchmark	Component count			Runtime (s)		Effective area (mm^2)		Linear channel length (mm)	
	IO	Valve	Other	OpenMFDA	Columba	OpenMFDA	Columba	OpenMFDA	Columba†
ChIP 4	42	51	0	4	1	255	862	489	114
ChIP 10	60	101	1	14	2	554	1107	1107	172
mRNAiso 4	30	59	0	6	1	294	1564	560	147
mnacidpro 3	27	51	0	5	1	255	5455	469	77

TABLE II

COMPARISON WITH COLUMBA-S BENCHMARK RESULTS [13].

† COLUMBA REPORTED FLUID CHANNEL LENGTHS ONLY. OPENMFDA RESULTS INCLUDE CONTROL CHANNEL LENGTHS.

Benchmark	Component count			Runtime (s)		Effective area (mm^2)		Linear channel length (mm)	
	IO	Valve	Other	OpenMFDA	Fluigi	OpenMFDA	Fluigi	OpenMFDA	Fluigi
simple	5	2	1	2	5	15	144	11	-
grad_cells	7	6	14	2	37	95	555	95	-
multi_input	11	4	5	2	25	40	424	37	-
hasty	5	0	22	3	22	110	285	162	-
rotary_cells	18	17	7	3	56	120	594	162	-
logic04	41	37	4	3	94	185	780	268	-
rotary16	53	65	0	3	161	324	985	503	-
net_mux	17	34	15	4	167	245	635	350	-
grid_8	30	134	78	4	769	739	707	522	-

TABLE III

COMPARISON WITH FLUIGI BENCHMARK RESULTS [23]. CHANNEL LENGTH WAS UNREPORTED.

5) 3D model generation - "tapeout": The layout is read into the 3D model generator tool. Structures and channels are generated following our proposed methodology and the resulting geometry is stored in the OpenSCAD language. The device geometry is then rendered into the standard STL 3D model format.

6) Slicing: The STL file is then sliced. The slicer tool evaluates a cross section of the 3D model on the $\hat{\mathbf{x}}\hat{\mathbf{y}}$ plane. The depth of each slice specifies the print layer depth. A 2D print layer mask image is generated for the slice, as well as a record specifying the print layer characteristics. Currently the tool generates a shared print layer depth and exposure time for each layer.

7) Printing and testing: The sliced design is then printed with an experimental high resolution SLA 3D printer [4]. Once the print is complete, uncured resin is flushed with an automated flushing system interfaced with the interconnect matrix.

V. RESULTS

A. Case study - urinalysis assay

To test the proposed methodologies and the design flow implementation, a benchmark device was implemented. The design performs a urinalysis assay for Ca^{2+} provided by a potential user based on a commercial assay that uses the arsenazo reagent. In this assay, the urine sample, the diluent, and the reagent (arsenazo) are mixed in a ratio of 2:225:25 respectively, and then optically measured using absorbance. The design consists of four serpentine channels and two mixers. OpenMFDA was used to generate the device from the hardware specification, as shown in Figure 3. The design was simulated using electrical analogs in SPICE. The device was then printed on an experimental SLA 3D printer [4]. The printed device was tested for functionality with an automated microfluidic test station.

B. Comparative benchmarks

Tables II and III provide comparative results for the benchmarks reported by Columba [13] and Fluigi [23]. To compare performance of MFDA tools, we follow the measurement and benchmarking methodology proposed in [26].

Benchmarks were run on an Intel Xeon Gold 6330N CPU at 2.20GHz. Runtimes for both Columba and OpenMFDA were similarly short. While the exact attention span of a typical user can only be speculated, providing results within 5-15 seconds is likely to meet the needs of practitioners who value quick feedback [1]. The runtime for Fluigi increases significantly with the size of the device, with run times measured in minutes for non-trivial devices. For all three tools, runtime scales with the size of the device. It is unclear from the Columba results whether the reported times represent the complete flow to final photomask or just the synthesis and PnR calculations.

Runtime numbers presented for OpenMFDA are for the execution of PnR and the generation of the device SCAD model. OpenSCAD does two rendering modes - a preview and a full geometric computation. The preview render updates in near real-time, which gives quick feedback to the user. The full computation to generate the accurate rendering for printing will take longer, between 30 seconds to 3 minutes. As the complexity of the geometry increases, the render time increases significantly. This is expected to only be done in preparation for printing, but does represent a potential bottleneck in the design flow.

Dimensional comparisons were made for area and linear channel length results. All tools use similar sized channels of approximately $30\mu m$, which indicates a common feature scale for size. To compare a device with multiple routing layers to a planar device, the effective area is estimated by multiplying the area of the device by the number of routing layers. Results are shown for a device with six routing layers. Area was minimized significantly, between 2 to 20 times smaller than

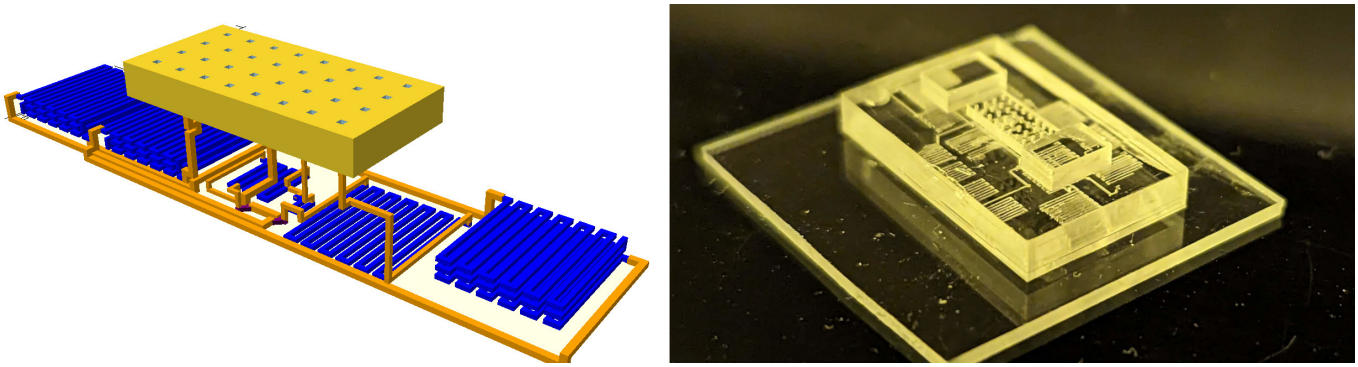

Fig. 3. (a) Rendered output of an arsenazo reagent assay device (negative space shown). Channels are shown in orange, serpentine in blue, mixers in purple, and the external interconnect matrix in yellow. (b) A printed device.

previous results. The exception is the grid_8 benchmark from Fluigi, which is a dense grid of valves connected directly to the neighboring components. This has very little routing, and the effective area metric disadvantages the 3D tool due to the number of unused routing layers. Experiments with the number of routing layers showed that increasing the number of layers to 16 reduced the linear channel length by 4-9%. This came at the cost of increased flow runtime. Increasing the number of routing layers also has consequences for manufacturing, as print time is a function of device depth. A routing solution was possible with as few as three layers.

Linear channel length is difficult to compare with the results reported by other tools. Fluigi did not report channel lengths in their results, preventing any comparison. Columba specifically only reports length of the fluid channels, without any measurements of control channel lengths. This significantly underrepresents the dimensions of the Columba results, and limits the ability to compare. 3D printed devices are not bound by the same constraint on physical separation of channel types, and OpenMFDA does not treat the two types differently. However, fluid volumes are likely more critical to the designer to ensure correct function and timing of the system, so reporting the two values separately does provide information of potential interest. Specifying and meeting design constraints will be of critical importance to the adoption of MFDA tools.

VI. Discussion

A. Comparison with existing MFDA methodologies

Direct comparisons between the OpenMFDA design flow and other tools discussed is difficult due to the use of a different manufacturing technology and different problem formulation. Much of the prior work in the field of MFDA has focused on addressing the limitation of a single fluid layer and a control layer. Our proposed methodology bypasses that constraint entirely. Formulating the MFDA PnR problem in terms of an analog of EDA PnR allows leveraging existing well developed EDA methods and tools. EDA is a mature field, and existing tools are capable of dealing with designs containing billions of transistors. All three methodologies compared are focused on the physical design processes, specifically PnR and manufacturable design generation. Columba provides capability to synthesize a reduced control line scheme, a significant feature lacking in Fluigi and OpenMFDA. However, it is also highly specialized to a specific class of fluidic devices, and is limited in extensibility to other designs.

B. Future directions

Our proposed design flow and the other flows are direct analogs to traditional EDA practices. However, practitioners are not digital designers, and further research and implementation will be required to understand the development methodologies and tools needed by practitioners [1].

A significant limitation of the proposed methodology is that formulation of the placement problem is still constrained in two dimensions. The SLA printing area is set by the available masking window size - this presents problems with scalability as the number of components increases. As seen in Tables II and III, the number of components in a particular benchmark was at most 211. Compared to gates in digital designs, each component represents a significant percentage of the available area. Conversely, depth of print is variable and much larger. This is a major departure in the analogy from the EDA problem. Placement of components in 3D is a possibility to leverage this available space.

VII. Conclusion

In this work, we have proposed a methodology for PnR and generating a 3D printed device. We created a formulation analogous to the standard EDA PnR problem to leverage existing EDA algorithms and tools. We presented our design flow OpenMFDA which implements the methodology. OpenMFDA has fast run-times and produces devices with significantly smaller area than a conventional PDMS-based device. Our contribution marks a significant step towards democratizing practical 3D printed microfluidics chips, making it more accessible to a broader range of researchers and practitioners, accelerating innovation in microfluidics applications. OpenMFDA is available as open-source software on Github at https://github.com/utah-mfda/openmfda_flow

REFERENCES

[1] J. McDaniel, W. H. Grover, and P. Brisk, "The case for semi-automated design of microfluidic very large scale integration (mVLSI) chips," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, ISSN: 1558-1101, 2017, pp. 1793–1798.

[2] J. McDonald, D. Duffy, *et al.*, "Fabrication of microfluidic systems in poly(dimethylsiloxane)," English, *Electrophoresis*, vol. 21, no. 1, pp. 27–40, 2000, ISSN: 0173-0835.

[3] T.-M. Tseng, M. Li, *et al.*, "Cloud columba: Accessible design automation platform for production and inspiration: Invited paper," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, ISSN: 1558-2434, 2019, pp. 1–6.

[4] G. P. Nordin, H. Gong, *et al.*, "3D printing for lab-on-a-chip devices with 20 um channels," in *Emerging Digital Micromirror Device Based Systems and Applications XI*, vol. 10932, SPIE, 2019, pp. 29–36.

[5] R. Sanka, K. Subacius, *et al.*, "Fluigi cloud - a cloud CAD platform for microfluidics," 2017, Accepted: 2017-11-15T15:12:24Z Publisher: IWBDA.

[6] J. K. Nunes and H. A. Stone, "Introduction: Microfluidics," *Chemical Reviews*, vol. 122, no. 7, pp. 6919–6920, 2022.

[7] J. McDaniel, B. Crites, *et al.*, "Flow-layer physical design for microchips based on monolithic membrane valves," *IEEE Design & Test*, vol. 32, no. 6, pp. 51–59, 2015, Conference Name: IEEE Design & Test, ISSN: 2168-2364.

[8] Q. Wang, H. Zou, *et al.*, "Physical co-design of flow and control layers for flow-based microfluidic biochips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 6, pp. 1157–1170, 2018, Conference Name: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, ISSN: 1937-4151.

[9] Y. Zhu, X. Huang, *et al.*, "Multicontrol: Advanced control-logic synthesis for flow-based microfluidic biochips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2489–2502, 2020.

[10] J.-L. Wu, K. S.-M. Li, *et al.*, "Solar: Simultaneous optimization of control-layer pins placement and channel routing in flow-based microfluidic biochips," in *2018 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 2018, pp. 1–4.

[11] Q. Wang, S. Zuo, *et al.*, "Hamming-distance-based valve-switching optimization for control-layer multiplexing in flow-based microfluidic biochips," in *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2017, pp. 524–529.

[12] H. Yao, T.-Y. Ho, and Y. Cai, "Pacor: Practical control-layer routing flow with length-matching constraint for flow-based microfluidic biochips," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6.

[13] T.-M. Tseng, M. Li, *et al.*, "Columba s: A scalable co-layout design automation tool for microfluidic large-scale integration," in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18, New York, NY, USA: Association for Computing Machinery, 2018, pp. 1–6, ISBN: 978-1-4503-5700-5.

[14] Q. Li, "Programming language development for microfluidic design," Technical University of Munich, Bachelor's Thesis, 2019.

[15] R. Sanka, J. Lippai, *et al.*, "3duF - interactive design environment for continuous flow microfluidic devices," *Scientific Reports*, vol. 9, no. 1, p. 9166, 2019, Number: 1 Publisher: Nature Publishing Group, ISSN: 2045-2322.

[16] J. Friend and L. Yeo, "Fabrication of microfluidic devices using polydimethylsiloxane," *Biomicrofluidics*, vol. 4, no. 2, p. 026 502, 2010, ISSN: 1932-1058.

[17] A. Bernardini, C. Liu, *et al.*, "Efficient spanning-tree-based test pattern generation for programmable microfluidic devices," *Microelectronics Journal*, vol. 79, pp. 38–45, 2018, ISSN: 0026-2692.

[18] A. Kafle, E. Luis, *et al.*, "3D/4D Printing of Polymers: Fused Deposition Modelling (FDM), Selective Laser Sintering (SLS), and Stereolithography (SLA)," *Polymers*, vol. 13, no. 18, p. 3101, 2021, ISSN: 2073-4360.

[19] J. L. Sanchez Noriega, N. A. Chartrand, *et al.*, "Spatially and optically tailored 3D printing for highly miniaturized and integrated microfluidics," en, *Nature Communications*, vol. 12, no. 1, p. 5509, 2021, Number: 1 Publisher: Nature Publishing Group, ISSN: 2041-1723.

[20] H. Gong, B. P. Bickham, *et al.*, "Custom 3D printer and resin for 18 um × 20 um microfluidic flow channels," en, *Lab on a Chip*, vol. 17, no. 17, pp. 2899–2909, 2017, Publisher: The Royal Society of Chemistry, ISSN: 1473-0189.

[21] T. Ajayi, V. A. Chhabria, *et al.*, "Toward an open-source digital flow: First learnings from the openroad project," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, ISBN: 9781450367257.

[22] H. Gong, A. T. Woolley, and G. P. Nordin, "3d printed high density, reversible, chip-to-chip microfluidic interconnects," en, *Lab on a Chip*, vol. 18, no. 4, pp. 639–647, 2018, ISSN: 1473-0189.

[23] H. Huang and D. Densmore, "Fluigi: Microfluidic device synthesis for synthetic biology," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 11, no. 3, 26:1–26:19, 2015, ISSN: 1550-4832.

[24] M. Kintel and C. Wolf, *OpenSCAD*.

[25] *Xyce(™) parallel electronic simulator*, [Computer Software] https://doi.org/10.11578/dc.20171025.1421, 2013.

[26] A. Snelgrove, S. Stockham, and P.-E. Gaillardon, "Benchmarking microfluidic design automation flows," in *2024 IFIP/IEEE 32nd International Conference on Very Large Scale Integration (VLSI-SoC)*, ISSN: 2324-8440, 2024, pp. 1–6.