Sortex: Efficient Timing-Driven Synthesis of Reconfigurable Flow-Based Biochips for Scalable Single-Cell Screening

Mohamed Ibrahim[†], Aditya Sridhar[†], Krishnendu Chakrabarty[†], and Ulf Schlichtmann[‡]

[†]Duke University, Durham, NC 27708, USA
[‡]Technical University of Munich, 80333 Munich, Germany

***Abstract*—Single-cell screening is used to sort a stream of cells into clusters (or types) based on pre-specified biomarkers, thus supporting type-driven biochemical analysis. Reconfigurable flow-based microfluidic biochips (RFBs) can be utilized to screen hundreds of heterogeneous cells within a few minutes, but they are overburdened with the control of a large number of valves. To address this problem, we present a pin-constrained RFB design methodology for single-cell screening. The proposed design is analyzed using computational fluid dynamics simulations, mapped to an RC-lumped model, and combined with a high-level synthesis framework, referred to as Sortex. Simulation results show that Sortex significantly reduces the number of control pins and fulfills the timing requirements of single-cell screening.**

***Index Terms*—Biochips, design automation, flow-based microfluidics, RC model, single-cell analysis, timing-aware synthesis.**

I. Introduction

Reconfigurable flow-based microfluidic biochips (RFBs) enable single-cell analysis by providing unprecedented fluid-handling capabilities for nanoliter droplets transported in a flow medium [1]. This technology allows analytical operations to be reduced to length scales on the order of single cells and be reconfigured during run-time. Using a microfluidic system, a stream of heterogeneous cells can be screened in a high-throughput manner, whereby every cell can be individually encapsulated into an aqueous droplet, incubated with a secreted enzyme-reporter molecule, sorted using a fluorescence sensing [2], and barcoded for single-cell analysis [3].

Flow-based microfluidics is today a commonly used technology for microbiology-on-chip; thousands of integrated valves can be used to manipulate droplets in a complex network of channels [4]. Advances in fabrication techniques have motivated design automation ("synthesis") research. Objectives of synthesis methods thus far have included: architectural-level design [5], efficient control mechanisms for on-chip valves [6], and reduction of the number of interfacing pins [7]. However, these methods cannot tackle the complexities of RFBs designed for high-throughput single-cell screening because of the following challenges:

(1) Scalable valve control: A large number of valves are needed to sort and barcode several cell types. These valves also need to be controlled in real-time in response to identified cell types; the control sequences for these valves are not known *a priori*. In [8], an RFB solution was introduced for scalable barcoding using a fully reconfigurable valve-based crossbar, where an n-to-m crossbar can route a barcoding droplet from any of the n inputs to any of the limited m outputs. However, it can be shown that a simple 40-to-4 crossbar requires at least 1344 valves, thus direct-addressing of valves using pressure ports is prohibitively expensive.

(2) Impact of chip parameters on performance: Parameters such as channel diameter and elasticity of the deformable membranes significantly influence the performance of the microfluidic application [9], [10]. For example, consider a microfluidic channel as depicted in Fig. 1(a). We carried out computational fluid dynamics (CFD) simulations using COMSOL [11] to characterize laminar flow at the channel output when a pressure step function is applied to the input. As shown in Fig. 1(b), the steady-state pressure at the output decreases as the input pressure and the channel diameter D decrease. Next, we assume that this channel is connected to a 90-kPa pressure source and it is used to actuate a valve that requires at least a 2-kPa pulse to deform the membrane (highlighted using a red line in Fig. 1(b)). We find that reliable valve actuation can be achieved only if D is larger than 2 mm; see Fig. 1(b). Similarly, by analyzing the flow rate of this pressure-driven mechanism (Fig. 1(c)), we find that the flow rate decreases as D decreases (Fig. 1(d)).

Therefore, neglecting the correlation between chip parameters and single-cell screening efficiency will invariably lead to unacceptable screening rates or even unexpected behaviour. It is important to investigate the delay associated with pressure-driven fluid transport to precisely predict the timing characteristics of a screening biochip. The result of this analysis can be utilized as a *latency constraint* to synthesize a biochip that is free of timing violations.

In this paper, we address the above challenges by investigating the design of a pin-constrained RFB for scalable single-cell screening. We utilize the delay model of pressure-driven transport to design a cost-effective RFB with high throughput. The contributions of this paper are as follows:

- We present a formal definition for the architecture of an RFB that is used for high-throughput single-cell screening.
- We describe a pin-sharing scheme that offers real-time multiplexed control of valves. An RC model is derived to describe the delay of the multiplexed control path and used to characterize chip performance. This model is validated using CFD simulations.
- We describe and evaluate a timing-driven synthesis solution for the control of RFBs. The solution (referred to as Sortex: cell sorter using multiplexed control) reduces the number of pins subject to performance constraints associated with single-cell screening.

The rest of the paper is organized as follows. Section II describes prior work and an RFB architecture for single-cell screening. In Section III, we introduce multiplexed control and the RC model. Next, Section IV presents the timing-driven synthesis technique. Experimental evaluation is presented in Section V and conclusions are drawn in Section VI.

II. Preliminaries

In this section, we review synthesis techniques for flow-based biochips and describe an RFB for single-cell screening.

A. Synthesis of Flow-Based Biochips

A typical flow-based microfluidic device is composed of two elastomer layers, as shown in Fig. 2(a). The layers are bonded using a sandwiched polydimethylsiloxane (PDMS) membrane, which forms a valve that controls fluid flow through the channels of the top layer [12]. Biochemical fluids are carried at the top layer (flow layer)

978-1-5386-3093-8/17/$31.00 ©2017 IEEE

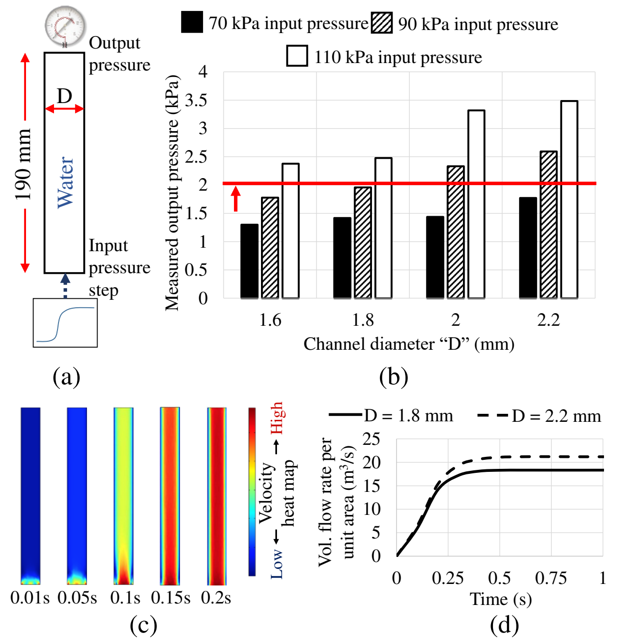

Fig. 1: The correlation between the parameters of a microfluidic channel and system performance: (a) geometric specifications, (b) evaluation of output pressure, (c) velocity profile of pressure-driven mechanisms, (d) dependence of flow rate on D.

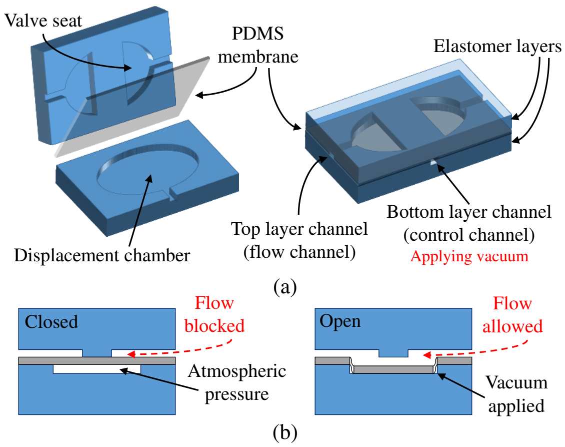

Fig. 2: (a) The components, and (b) the operation of a normally closed valve [12].

and the other layer (control layer) provides the vacuum to deflect the PDMS membrane "outside" the flow channel to permit fluid flow–this valve is characterized as "normally closed" (Fig. 2(b)). Synthesis techniques have been developed for channel placement and routing in each layer separately [5], [13], [14]. These methods, however, overlooked the interaction between the control and flow layers, thus they may lead to infeasible solutions. The interaction between the control and flow layers was highlighted in [15], and solutions for pin-count reduction were presented in [6], [7]. These methods, however, are inadequate for high-throughput screening for the following reasons:

(***i***) For control-pin minimization, [7] relies on activation-based compatibility; valve-actuation patterns of a protocol are mapped to a pin-count minimization strategy. A more flexible approach in [6] explores compatibility among basic control actions of individual fluidic operations. However, both techniques assume that "compatible" valves can be simultaneously addressed using the same pressure source—this assumption is valid only with a small number of compatible valves due to fan-out limits. Moreover, these approaches do not enable independent actuation of valves, limiting reconfigurability for single-cell screening.

(***ii***) The work in [6] considered pressure-propagation delay within fluidic-channel routing using a *path-length* model. However, this model does not capture the correlation between fluid dynamics and biochip parameters (e.g., channel width and elasticity). Furthermore, [6] uses only the longest pressure-propagation delay of a pin-sharing valve group to assess performance. This approach cannot be used with multiplexed control of independently addressable valves.

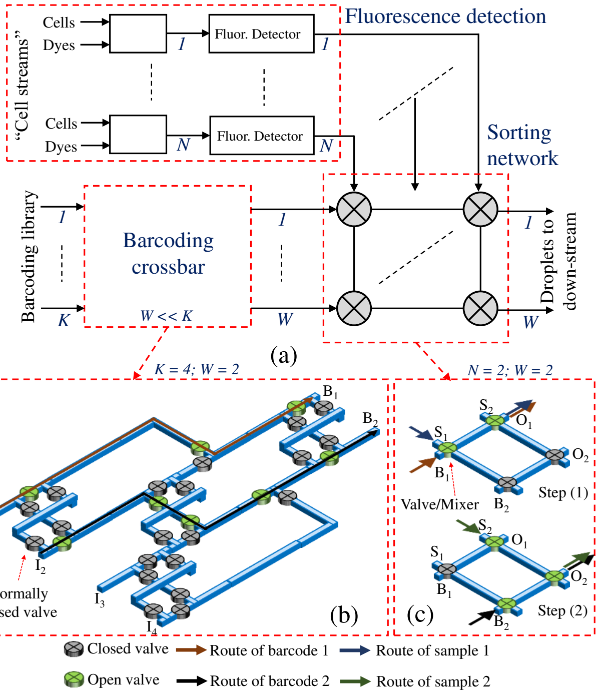

Fig. 3: An RFB for single-cell screening: (a) platform modules, (b) a 4-to-2 barcoding module [8], (c) a 2-by-2 sorting network [1].

B. An RFB for Single-Cell Screening

Fig. 3(a) shows an RFB architecture that screens N streams of cells, classifies cells into K types, and barcodes the cells via W ports ($W << K$). This platform consists of three modules: valve-less fluorescence detection, barcoding crossbar, and sorting network. Adaptation is achieved via the detection of samples in the fluorescence-detection module, whereas reconfiguration is carried out *in response* at both the barcoding crossbar and the sorting network.

A reconfigurable valve-based crossbar is employed to barcode droplets and route them towards the sorting network. A barcoding droplet is routed from an input port $I_k \in \{I_1, I_2, ..., I_K\}$ to an output port $B_w \in \{B_1, B_2, ..., B_W\}$ through channels, and the routing path is configured online using a set of valves $\mathcal{V}^p = \{V_1, V_2, ..., V_P\}$; P is the number of crossbar valves. Fig. 3(b) shows a 4-to-2 crossbar that routes two different barcoding droplets concurrently.

Connected to the crossbar is a sorting module that mixes a sample droplet from a stream $S_n \in \{S_1, S_2, ..., S_N\}$ with a barcoding droplet generated from B_w. This function can be implemented using $N \times W$ independent mixers. However, this approach makes channel network design overly complex for large N and W. Scalable single-cell sorting requires a programmable microfluidic platform that performs biochemical operations at functional units *on-the-fly*. We exploit the programmable microfluidic network proposed in [1], where an N-by-W network can dynamically process any pair of $N \times W$ input droplets. A set of valves $\mathcal{V}^q = \{V_1, V_2, ..., V_Q\}$, where Q is the number of network valves, are used to mix barcodes with cells and route mixed droplets to output ports in the set $\{O_1, O_2, ..., O_W\}$. Fig 3(c) illustrates a 2-by-2 network.

While this RFB design offers reconfigurability, the efficiency of single-cell screening depends on the topology of the flow channels and how fast the flow valves in the set $\mathcal{V} = \mathcal{V}^p \cup \mathcal{V}^q$ can be actuated. Since we consider a fixed topology for the flow channels, we focus on delays associated with valve actuation (pressurization or depressurization). We demonstrate in the next section that an effective pin-constrained control methodology for an RFB must consider the timing overhead of valve actuation. Therefore, we consider valve-based modules (the barcoding crossbar and the sorting network) to investigate the sequence and delay of valve controls required to route a barcode from $I_k : \{k \in \mathbb{N}, k \leq K\}$ to $O_w : \{w \in \mathbb{N}, w \leq W\}$ and to route the barcoded cell from $S_n : \{n \in \mathbb{N}, n \leq N\}$ to $O_w : \{w \in \mathbb{N}, w \leq W\}$. This sequence is referred to as a *control sequence* and is denoted by $\boldsymbol{H_t} : \{\boldsymbol{H_t} \in \{0,1\}^X, t \in \mathbb{N}, t \leq T\}$, where X is the number of chip valves ($X = P+Q$) and T is the total number of possible control sequences. The parameter T can also be interpreted as the number of possible flow paths that can be utilized by a single-cell sample of any type. In addition, we use $\theta(X, Y)$ to denote a screening biochip that contains X valves and is actuated using Y control pins.

Our goal in this paper is to provide an architectural-level synthesis scheme that allows the control of X valves by Y pins ($Y << X$), while the latency of every control sequence $\boldsymbol{H_t}$ is less than a threshold. A solution to this problem leads to a biochip that provides a desired screening throughput.

III. Multiplexed Control and Delay

In this section, we explain the pin-constrained design methodology and the associated delay model.

A. Multiplexed Control

To reduce the number of control pins, we allow several valves in a screening biochip to share a few pins using *time-division multiplexing* (TDM) [12]. Fig. 4(b) shows an example of multiplexed control for an 8-valve flow channel (shown in Fig. 4(a)); i.e., $T = 1$. As shown in Fig. 4(b), the circles in blue and orange represent the control pins of the biochip whereas the circles in white represent the valves.

To implement TDM, two types of control pins are needed: (1) A set of primary pins $\mathcal{C}^a$ (blue circles), used to provide the pressure (or vacuum) through primary control channels to actuate valves in the flow channel; (2) a set of demultiplexing pins $\mathcal{C}^g$ (orange circles), used to direct the pressure-driven flow from a primary pin to a particular valve in the flow channel through demultiplexing control channels, thereby allowing flow valves to be independently addressable. The pins in $\mathcal{C}^g$, configured as two-way sources [12], can pressurize and de-pressurize a set of valves, referred to as control valves $\mathcal{L}$ (Fig. 4(b)) to avoid confusion with the flow valves in Fig. 4(a). A control valve $l_i \in \mathcal{L}$ acts as a two-way switch to direct a control pulse in one of two directions.

For example, to actuate flow valve V_2 in Fig. 4(a), demultiplexing pins C_1^g, C_2^g, and C_3^g (shown in Fig. 4(b)) are first activated to switch the control valves l_7, l_1, and l_5, respectively. By switching these control valves, a control path is now opened between the primary pin C_1^a and the desired flow valve V_2. Next, C_1^a is actuated to open or close V_2, which in turn is a latching valve that can maintain its open or closed state while disconnected from the controller [12].

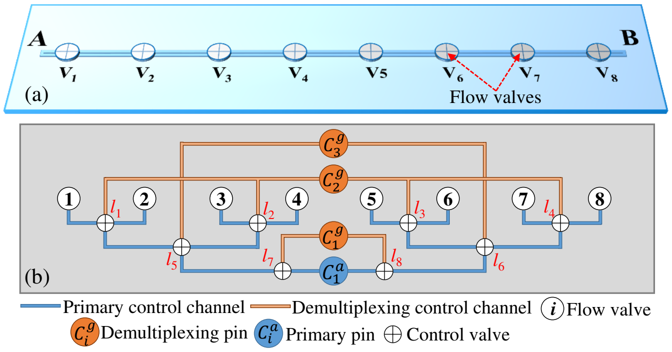

Fig. 4: (a) An 8-valve channel; (b) Mutliplexed control of the channel using 4 control pins.

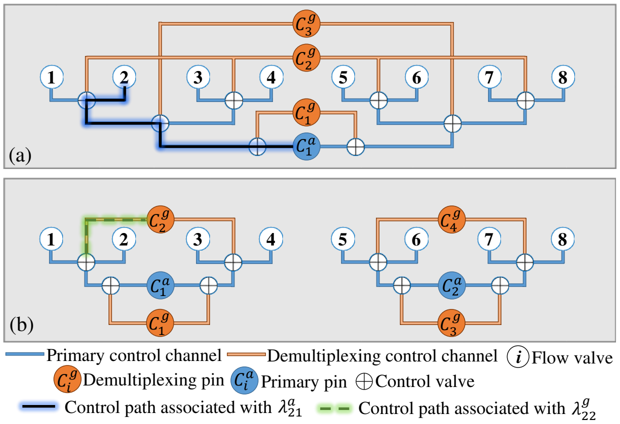

Fig. 5: Multiplexed control of an 8-valve channel using (a) 4 control pins and (b) 6 control pins.

Note that we can route a fluid through the flow channel in Fig. 4(a) from A to B or vice versa only after all the 8 valves are actuated. In addition, the flow circuitry in Fig. 4(a) and the multiplexed control circuitry in Fig. 4(b) are located on different layers.

By using multiplexed control and by considering the valve-control signal as a binary signal, X flow valves can be independently actuated using only a single primary pin and $\lceil \log_2 X \rceil$ demultiplexing pins; i.e., $Y = \lceil \log_2 X \rceil + 1$ provides a lower bound on the number of control pins needed to actuate X valves. For example, the 8 valves in Fig. 4(a) can be addressed using only 4 control pins. Similarly, the valves of an 1024-valve biochip can be addressed using 11 pins.

Note however that the addressing of a single flow valve requires a sequence of pin actuations, which imposes timing overhead. For the 8-valve channel in Fig. 4(a), Fig. 5(a-b) illustrates multiplexed control for actuating all the channel valves using 4 pins and 6 pins, respectively. Although Fig. 5 does not show the delays of control pulses through the control circuitry (the delays are expected to be larger for the 4-pin design), it is evident that the multiplexed control procedure is less complicated with 6 control pins. Hence, there is a tradeoff between pin-count reduction and the complexity of multiplexed control, and therefore the performance of single-cell screening.

Consider a screening biochip $\theta(X, Y)$ with multiplexed control. Also, consider a variable d^i_{xy} that specifies the delay contributed by a pin C^i_y of type i to actuate a valve V_x. Let λ^a_{xy} be a connectivity function such that $\lambda^a_{xy} = d^a_{xy}$ if a primary pin $C^a_y \in \mathcal{C}^a$ is used to actuate a flow valve $V_x \in \mathcal{V}$ with actuation delay d^a_{xy}, and $\lambda^a_{xy} = 0$ otherwise. Likewise, $\lambda^g_{xy} = d^g_{xy}$ indicates that a demultiplexing pin $C^g_y \in \mathcal{C}^g$ is used in the actuation of V_x with delay d^g_{xy}.

Definition 1. *The* connectivity vector $\boldsymbol{\phi_x} \in \mathbb{R}^Y$ *of a valve* V_x *is defined as* $\boldsymbol{\phi_x} = [\lambda^a_{x1}\ \lambda^a_{x2}\ \ldots\ \lambda^a_{xA}|\lambda^g_{x1}\ \lambda^g_{x2}\ \ldots\ \lambda^g_{xG}] = [\boldsymbol{\phi^a_x}|\boldsymbol{\phi^g_x}]$, *where* A *and* G *are the number of primary pins and demultiplexing pins, respectively* ($Y = A + G$), $\boldsymbol{\phi^a_x}$ *is the connectivity associated with the primary pins, and* $\boldsymbol{\phi^g_x}$ *is the connectivity associated with the demultiplexing pins. In addition, the connectivity matrix for* $\theta(X, Y)$ *is defined as* $\boldsymbol{\Phi} = [\boldsymbol{\Phi^a}|\boldsymbol{\Phi^g}] = [\boldsymbol{\phi_1}\ \boldsymbol{\phi_2}\ \ldots\ \boldsymbol{\phi_X}]^\intercal$*; where* $\boldsymbol{\Phi^a}$ *and* $\boldsymbol{\Phi^g}$ *are the connectivity matrices associated with the primary pins and demultiplexing pins, respectively.* □

We use the multiplexed control designs in Fig. 5(a-b) for explanation. In Fig. 5(a), the actuation of any flow valve $V_x \in \{V_1, V_2, ..., V_8\}$ is accomplished using the same primary pin C^a_1 and the same three demultiplexing pins $\{C^g_1, C^g_2, C^g_3\}$. As a result, $\boldsymbol{\phi_x} = [\lambda^a_{x1}|\lambda^g_{x1}\ \lambda^g_{x2}\ \lambda^g_{x3}], 1 \le x \le 8$. Fig. 5(a) depicts the control path associated with λ^a_{21}. On the other hand, in Fig. 5(b), the actuation of valves $\{V_1, V_2, V_3, V_4\}$ is accomplished using primary pin C^a_1 and demultiplexing pins $\{C^g_1, C^g_2\}$, whereas valves $\{V_5, V_6, V_7, V_8\}$ are addressed using primary pin C^a_2 and demultiplexing pins $\{C^g_3, C^g_4\}$. Therefore, $\boldsymbol{\phi_x}$ is defined in this case as follows:

$$\boldsymbol{\phi_x} = \begin{cases} [\lambda^a_{x1}\ \ 0\ \ |\lambda^g_{x1}\ \lambda^g_{x2}\ \ 0\ \ \ 0\ ] & \text{if } 1 \le x < 5 \\ [\ 0\ \ \ \lambda^a_{x2}|\ 0\ \ \ 0\ \ \ \lambda^g_{x3}\ \lambda^g_{x4}] & \text{if } 5 \le x \le 8 \end{cases}$$

Fig. 5(b) depicts the control path associated with λ^g_{22}. Note that only a single primary pin can be used in the actuation of a certain flow valve (i.e., V_x cannot be addressed by both C^a_1 and C^a_2). Hence, flow valves are divided into groups; each group is associated with a particular primary pin. To prevent control interference between groups, each demultiplexing pin can only be used in the actuation of flow valves that belong to the same group. By this grouping method, different groups of flow valves can be actuated concurrently.

While the flow valves $\{V_1, V_2, ..., V_8\}$ construct only a single control sequence (i.e., $T = 1$), the following definition considers the generic case where $T > 1$.

Definition 2. *A* control sequence $\boldsymbol{H_t} : \{t \in \mathbb{N}, t \le T\}$ *is a vector of length* X *such that* $\boldsymbol{H_t}[x] = 1$ *indicates that a flow valve* V_x *must be actuated as a part of the control sequence, and* $\boldsymbol{H_t}[x] = 0$ *otherwise. In addition, the control matrix of* $\theta(X, Y)$ *is defined as* $\boldsymbol{\Gamma} = [\boldsymbol{H_1}\ \boldsymbol{H_2}\ \ldots\ \boldsymbol{H_T}]^\intercal$. □

The simplest form of an RFB that contains more than a single control sequence is a 2-by-2 crossbar (also known as a transposer [8]); see Fig. 6. Such a transposer consists of six valves $\{V_1, V_2, ..., V_6\}$ that can be configured to route a biochemical sample through four different paths ($T = 4$): I_1 to B_1, I_1 to B_2, I_2 to B_1, and I_2 to B_2. To activate the second path (i.e., I_1 to B_2), valves $\{V_2, V_4\}$ must be opened and valves $\{V_1, V_6\}$ must be closed. Hence the control sequence $\boldsymbol{H_2}$ is defined as follows: $\boldsymbol{H_2} = [1\ 1\ 0\ 1\ 0\ 1]$. Note that a pressure pulse for closing a valve and a vacuum pulse for opening the same valve are considered to have the same delay characteristics [12].

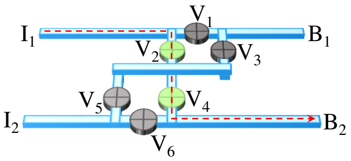

Fig. 6: A transposer design that exhibits 4 different flow paths (i.e., 4 control sequences) [8].

Next, we compute the latency associated with a control sequence $\boldsymbol{H_t}$. Recall that $\boldsymbol{\Phi}$ is an augmented matrix; $\boldsymbol{\Phi} = (\boldsymbol{\Phi^a}|\boldsymbol{\Phi^g})$, where $\boldsymbol{\Phi^a}$ and $\boldsymbol{\Phi^g}$ are the connectivity matrices associated with the primary pins and demultiplexing pins, respectively (Definition 1). By examining the multiplexed control procedure for each flow valve, we observe that the demultiplexing pins are used first to actuate the control valves, which in turn select the control path within the multiplexed control circuitry. Next, the primary pin is activated to actuate the target flow valve. With this procedure, demultiplexing pins can be activated concurrently, whereas a primary pin can be activated only after a control path is available. This observation leads us to the following lemma.

Lemma 1. *The control delay of a flow valve* V_x *is a value* $b_x = \max\{\boldsymbol{\phi^a_x}\} + \max\{\boldsymbol{\phi^g_x}\}$*, and the control delay vector for all the valves in* $\theta(X, Y)$ *is* $\boldsymbol{\beta} = [b_1\ b_2\ \ldots\ b_X]^\intercal$.

We use the multiplexed control example in Fig. 5(b) to illustrate the above lemma. Recall that $\boldsymbol{\phi_2} = [\boldsymbol{\phi^a_2}|\boldsymbol{\phi^g_2}] = [\lambda^a_{21}\ 0|\lambda^g_{21}\ \lambda^g_{22}\ 0\ \ 0]$. Since the demultiplexing pins $\{C^g_1, C^g_2\}$ are actuated concurrently, then the control delay is associated with the worst-case (i.e., maximum) delay among $\{\lambda^g_{21}, \lambda^g_{22}\}$, which is defined as $\max\{\lambda^g_{21}, \lambda^g_{22}, 0, 0\} = \max\{\boldsymbol{\phi^g_2}\}$. Since the primary pin C^a_1 is actuated after the control path is opened, then its associated delay is defined as $\lambda^a_{21} = \max\{\boldsymbol{\phi^a_2}\}$, and the effective control delay associated with V_2 is $b_2 = \max\{\boldsymbol{\phi^a_2}\} + \max\{\boldsymbol{\phi^g_2}\} = \lambda^a_{21} + \max\{\lambda^g_{21}, \lambda^g_{22}\}$.

Based on this key result, we can compute the latency of a control sequence $\boldsymbol{H_t}$, while taking into consideration the grouping of flow valves according to the assigned primary pins. The control valves that belong to the same group cannot be actuated concurrently since they share the same primary pin, but different valve groups can be actuated concurrently. The concurrent actuation of different valve groups (even those which belong to the same control sequence; see Fig. 5(b)) can be captured by considering only the worst-case delay among all the groups. For this purpose, we define the connectivity of a primary pin C^a_y as $\boldsymbol{\rho_y} = [\lambda^a_{1y}\ \lambda^a_{2y}\ \ldots\ \lambda^a_{Xy}]^\intercal \in \mathbb{R}^X$, $y \in [1, A]$. We use a binary vector $sign(\boldsymbol{\rho_y})$ to specify the set of flow valves that belong to group C^a_y.

Consider the vector operator $\circ$ that provides the element-wise product of two vectors, i.e., $\boldsymbol{E} \circ \boldsymbol{B} = [e_1\ e_2\ \ldots\ e_n] \circ [b_1\ b_2\ \ldots\ b_n] = [e_1b_1\ e_2b_2\ \ldots\ e_nb_n]$. Hence, the relation $sign(\boldsymbol{\rho_y}) \circ \boldsymbol{\beta}$ specifies the control delays associated with the flow valves in group C^a_y. A subset of the flow valves in C^a_y belongs to a control sequence $\boldsymbol{H_t}$ and therefore they need to be actuated serially if $\boldsymbol{H_t}$ is selected for single-cell screening. As a result, the relation $\boldsymbol{H_t} \cdot (sign(\boldsymbol{\rho_y}) \circ \boldsymbol{\beta})$ computes the cumulative control delay associated with a set of flow valves that is characterized as follows: (1) the flow valves belong to group C^a_y; (2) the actuation of these valves is needed for activating $\boldsymbol{H_t}$. By computing the cumulative control delays associated with all groups of flow valves within $\boldsymbol{H_t}$, we obtain the worst-case control delay among all groups as follows: $\max\limits_{1 \le y \le A} \{\boldsymbol{H_t} \cdot (sign(\boldsymbol{\rho_y}) \circ \boldsymbol{\beta})\}$. Based on this observation, we have derived the following theorem, which forms the basis for the Sortex synthesis procedure (Section IV).

Theorem 1. *If* α_t*, where* $\alpha_t \in \mathbb{R}, t \in \mathbb{N}, t \le T$*, is the cumulative* *control latency value associated with a control sequence* $\boldsymbol{H_t}$ *in a chip* $\theta(X,Y)$*, then* $\alpha_t = \max_{1 \le y \le A}\{\boldsymbol{H_t} \cdot (sign(\boldsymbol{\rho_y}) \circ \boldsymbol{\beta})\}$*, where* $\circ$ *is the element-wise product. In addition, the cumulative control latency vector for all control sequences in* $\theta(X,Y)$ *is* $\boldsymbol{\Theta} = \max_{1 \le y \le A}\{\boldsymbol{\Gamma} \cdot (sign(\boldsymbol{\rho_y}) \circ \boldsymbol{\beta})\}$*.*

Fig. 6 illustrates Theorem 1. Recall that the second control sequence is $\boldsymbol{H_2} = [1\ 1\ 0\ 1\ 0\ 1]$. First, assume that all six valves belong to the same group, i.e., they are actuated using the same primary pin C_1^a. In this case, $sign(\boldsymbol{\rho_1}) = [1\ 1\ \ldots\ 1]$. To activate $\boldsymbol{H_2}$, V_1, V_2, V_4, V_6 need to be actuated serially; therefore the cumulative control latency is the sum of the control delays associated with these valves. In other words, $\alpha_2 = \boldsymbol{H_2} \cdot (sign(\boldsymbol{\rho_1}) \circ \boldsymbol{\beta}) = [1\ 1\ 0\ 1\ 0\ 1] \cdot [b_1\ b_2\ \ldots]^\intercal = b_1 + b_2 + b_4 + b_6$. Second, if two primary pins $\{C_1^a, C_2^a\}$ are used, where $sign(\boldsymbol{\rho_1}) = [1\ 1\ 1\ 0\ 0\ 0]$ and $sign(\boldsymbol{\rho_2}) = [0\ 0\ 0\ 1\ 1\ 1]$, the cumulative control latency α_2 is computed as $\alpha_2 = \max_{1 \le y \le 2}\{[1\ 1\ 0\ 1\ 0\ 1] \cdot (sign(\boldsymbol{\rho_y}) \circ \boldsymbol{\beta})\} = \max\{b_1 + b_2,\ b_4 + b_6\}$.

The above discussion is focused on the latency of multiplexed control in an RFB. Since the topology of flow channels in the proposed RFB is fixed, we can easily estimate the flow latency of samples through these channels and therefore obtain an accurate estimate of biochip throughout. Note that a sample can be routed through a flow path only after the associated control sequence is activated. Therefore, if the flow latency vector associated with the biochip control sequences is defined as $\boldsymbol{\omega} \in \mathbb{R}^T$, then the effective latency vector of $\theta(X,Y)$ is $\boldsymbol{\Theta} + \boldsymbol{\omega}$ and the worst-case latency is $\tau = \max\{\boldsymbol{\Theta} + \boldsymbol{\omega}\}$. To optimize the throughput of single-cell screening, our synthesis method in Section IV optimizes the multiplexed control scheme in an RFB such that Y is minimized and $\tau \le \eta$, where η is a predefined value.

B. Delay of Pressure-Driven Fluid Transport

In electrical circuits, the delay of an electrical signal through a wire can be characterized using a delay model [16]; a widely used delay model, especially with wires characterized as RC trees or ladders, is the Elmore delay model [16].

In analogy with electrical circuits, the delay of laminar flow through a long elastic channel can be approximated using an equivalent Elmore delay model, which is a practical alternative to complex CFD simulations. For this purpose, there is a need to define the model components, i.e., the hydraulic resistance R and the hydraulic compliance M [10].

A laminar flow of a fluid through a long channel can be described using the Hagen-Poiseuille equation: $Q^H = \frac{\pi s^4 \cdot \Delta Pr}{8\mu \cdot b}$, where Q^H is the flow rate, s is the channel radius, $\Delta Pr = Pr_{in} - Pr_{out}$ is the pressure drop across the channel, μ is the dynamic viscosity of the fluid, and b is the length of the channel. The analog of this law in electrical circuits is Ohm's law. We use this analogy to estimate the *hydraulic resistance* R, which is defined as $R = \frac{\Delta Pr}{Q^H} = \frac{8\mu \cdot b}{\pi s^4}$.

The above model makes the assumption that a microfluidic channel is rigid. However, RFBs are fabricated using elastic material (e.g., PDMS) hence pressure can cause the cross-sectional area of a channel to change; see Fig. 7(a). The capability of an elastic channel to store fluid when pressurized is known as hydraulic compliance (similar to capacitance in electrical circuits). To verify this behavior, we conducted transient CFD simulation using COMSOL for laminar flow in a PDMS channel. Fig. 7(b) shows channel deformation at different time steps when a pressure step Pr_1 is applied to the input. This deformation can be interpreted as the potential of a channel to store fluid when pressurized; this is referred to as *hydraulic compliance* M. We also observe that the deformation distance and therefore M are correlated with the material elasticity (i.e., channel dilatability γ). This observation corroborates an analytic expression that specifies the hydraulic compliance for an elastic circular channel [17]: $M = \gamma \cdot \pi s^2 \cdot b$, where γ is the channel dilatability, s is the channel radius, and b is the channel length.

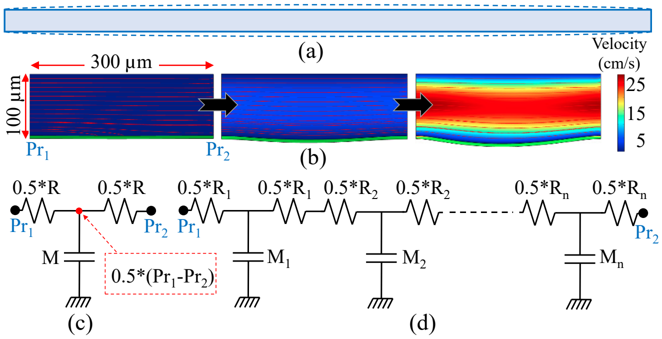

Fig. 7: (a) A PDMS microfluidic channel expands when pressurized. (b) Result of transient CFD simulation for an elastic channel under pressure. (c) Lumped-RC model. (d) Distributed-RC model.

The closed-form equations for R and M can be used to estimate fluidic delay. Furthermore, since the values of R and M are specified based on the channel geometry and material elasticity, the fluidic delay, and therefore the screening throughput, can be tuned based on these parameters. According to [18], a straightforward approach for modeling an elastic channel is by using a lumped-RC model, as shown in Fig. 7(c). For this model, the fluidic delay of the channel is simply $R \cdot M$. However, this lumped model does not take into account the significant change in pressure across the channel. To capture the variation in pressure and its impact on fluidic delay, we use a distributed-RC ladder, as shown in Fig. 7(d). This model offers three advantages: (1) increasing the number of model segments n enhances model accuracy since every segment exhibits an infinitesimal change in pressure; (2) similar to the models of electrical interconnects [16], modeling various segments of a channel provides the opportunity to design a width-varying channel—this design method can be employed to minimize both the fluidic delay and the channel area; (3) the delay associated with a distributed-RC ladder (containing n segments) can be estimated using the Elmore model [16]: $d_{el} = \sum_{i=1}^{n}\left(M_i \cdot (\sum_{j=1}^{i} R_j)\right)$, where d_{el} is the Elmore delay, R_j and M_j are the resistance and compliance of segment j, respectively. Hence, we adopt this model in our synthesis framework to estimate the delay associated with flow and multiplexed-control paths.

IV. Sortex: Synthesis Solution

In this section, we discuss the problem formulation and the proposed synthesis framework (Sortex).

A. Problem Formulation

We consider the following problem formulation in this work.

Input: (1) The specifications of a screening biochip $\theta(X,Y)$, which includes: *(i)* the locations of all the valves $\mathcal{V}$ and $\mathcal{L}$; *(ii)* the channels connecting the flow valves $\mathcal{V}$; *(iii)* the maximum number of control pins ($\mathcal{C}^a$ and $\mathcal{C}^g$) and their locations; *(iv)* possible values of the channel width $\mathcal{D} = \{D_1, D_2, \ldots\}$.
(2) A fluidic delay model based on a distributed RC ladder.
(3) A threshold $\eta \in \mathbb{R}$ that represents the maximum latency.

(4) An upper bound Ar_{max} on the channel area.

Output: (1) The screening matrix $\mathbf{\Gamma}$. (2) The connectivity matrix $\mathbf{\Phi}$. (3) The worst-case latency τ. (4) The dimensions of the control channels: lengths $\mathcal{U} = \{u_1, u_2, ...\}$ and widths $\mathcal{F} = \{f_1, f_2, ...\}$.

Constraints: (1) Screening latency constraint ($\tau \leq \eta$). (2) Area constraint ($\sum_i (u_i \times f_i) \leq Ar_{max}$).

Objective: Minimize the number of control pins Y used for multiplexed control of biochip valves.

Note that efficient channel routing is beyond the scope of this paper. Sortex calculates control delays by using Manhattan distances between different entities. This work can be extended further by incorporating routing as in [6].

B. Sortex Algorithm

Recall that flow valves are assigned to groups; each group is actuated using a dedicated primary pin. To reduce the number of control pins, synthesis for multiplexed control must increase the number of flow valves assigned to a group.

We use a heuristic method based on divide-and-conquer (Algorithm. 1). Initially, the proposed method selects a primary pin (Line 13) and iterates over the flow valves in a pairwise manner (Line 10) trying to connect every pair to the current primary pin (Line 16). When the multiplexed-control scheme is expanded, timing analysis is performed using the Elmore delay model from Section III-B and the relation from Theorem 1 to ensure that the latency constraint is not violated (Line 17). If no constraint violation is detected, the expanded multiplexed control is accepted and the associated valve/pin variables are updated (Lines 18-21). However, if a violation is detected, the proposed connection to the current primary pin is declined and a new primary pin is selected for connection (Lines 23-24). In this case, the flow valves are divided into two groups; one group combines the valves that have already been connected to the first primary pin and the second group combines the remaining valves (including the current pair)—we call this scheme *divide-and-conquer*. When the remaining flow valves are connected to a second primary pin, we further divide the set of valves into two groups according to the latency constraint. This process continues until all the flow valves are addressed.

The order of selection of the valves impacts computational performance. Random selection of valves may lead to high CPU time because distant valves can be combined in a single group, causing divide-and-conquer to be invoked frequently; i.e., unnecessarily increasing the number of control pins. To address this issue, we create a priority queue of the unaddressed flow valves (Line 2). The priority of valve selection is decided according to the following policy: (1) select a valve V_f randomly and place it at the head of the queue; (2) select a set of flow valves that does not belong to the same control sequences as V_f and sort these valves in an increasing order according to their Manhattan distance from V_f; (3) sort the remaining valves according to their Manhattan distance from V_f. This priority scheme (worst-case time complexity $O(T \cdot X + X \cdot \log X)$) is invoked whenever a divide-and-conquer action takes place (Line 25).

We next connect a new pair of flow valves to a primary pin (i.e., TestMC function in Algorithm 1). To expand the multiplexed control, control paths are designed to manage the actuation of the new flow valves. The synthesis of a control path is governed by two aspects: (1) the selection of control valves $\mathcal{L} = \{l_1, l_2, ...\}$ through which the control path is designed; (2) the selection of a new demultiplexing pin (if needed) in order to actuate the control valve. Consider the design of the control paths for the RFB in Fig. 8(a). To simplify the selection of control valves at each iteration, we initially split $\mathcal{L}$ into logical levels (Line 3), where the first level contains control valves that will be directly connected to the flow valves, e.g., l_1 in Fig. 8(b) and $\{l_1, l_2\}$ in Fig. 8(c). Such an organization is performed in advance and it enforces two conditions: (1) two control valves cannot be connected using primary control channels if they belong to the same logical level; (2) a single demultiplexing pin can only be used to actuate the control valves that belong to the same level and are used in the actuation of a single group of flow valves. For example, in Fig. 8(c), $\{l_1, l_2\}$ can be actuated using only C_1^g, which cannot be used to actuate l_3.

Algorithm 1 Sortex Procedure

```
1: $\Gamma \leftarrow$ ConstructScreeningMatrix($\mathcal{V}$);
2: $\mathcal{VQ} \leftarrow$ ConstructValvesPriorityQueue($\mathcal{V}$);
3: AssignControlValvesToLevels($\mathcal{L}$);
4: SortDecrChannelWidthsRange($\mathcal{D}$);
5: $j \leftarrow 0$; // Iterator
6: repeat
7:     $\mathcal{CC}^a \leftarrow \emptyset$; $\mathcal{CC}^g \leftarrow \emptyset$; // Connected pins so far
8:     $\mathcal{U} \leftarrow \emptyset$; $\mathcal{F} \leftarrow \emptyset$; $\Phi \leftarrow [0]$; $\tau \leftarrow 0$; $A \leftarrow 0$;
9:     $D_j \leftarrow$ ConsiderChannelWidth($\mathcal{D}$);
10:    for $(V_i, V_{i+1}) \in \mathcal{VQ}$ do
11:        $\mathcal{X} \leftarrow \emptyset$; $\mathcal{Y} \leftarrow \emptyset$; // Connected pins to the pair
12:        if $\mathcal{CC}^a = \emptyset$ then
13:            $\mathcal{X} \leftarrow$ selectNearestPrimaryPinLocation();
14:            $\mathcal{Y} \leftarrow$ BuildMC($V_i, V_{i+1}, \mathcal{X}$,"2-Pin");
15:        else
16:            $\{\Phi, \text{Conf}\} \leftarrow$ TestMC($V_i, V_{i+1}$);
17:            $\tau \leftarrow$ CalculateLatencyUsingElmore($\Phi, D_j$);
18:            if $\tau \leq \eta$ and Conf = "1-Pin" then
19:                $\mathcal{Y} \leftarrow$ BuildMC($V_i, V_{i+1}$, "1-Pin");
20:            else if $\tau \leq \eta$ and Conf = "0-Pin" then
21:                BuildMC($V_i, V_{i+1}$, "0-Pin");
22:            else
23:                $\mathcal{X} \leftarrow$ selectNearestPrimaryPinLocation();
24:                $\{\mathcal{X}, \mathcal{Y}\} \leftarrow$ BuildMC($V_i, V_{i+1}, \mathcal{X}$, "2-Pin");
25:                UpdatePriorityQueue($\mathcal{VQ}$);
26:        UpdateVariables($\mathcal{U}, \mathcal{F}, \Phi, \tau$);
27:        $\mathcal{CC}^a \leftarrow \mathcal{CC}^a \cup \mathcal{X}$; $\mathcal{CC}^g \leftarrow \mathcal{CC}^g \cup \mathcal{Y}$;
28:        $Ar \leftarrow$ CalculateChannelsArea($\mathcal{U}, \mathcal{F}$);
29:        if $Ar > Ar_{max}$ then break;
30:    $j \leftarrow j + 1$;
31: until $Ar \leq Ar_{max}$
32: return $\{\mathcal{U}, \mathcal{F}, \mathcal{CC}^a, \mathcal{CC}^g, \tau, \Phi\}$;
```

Based on the above hierarchy of control valves, the synthesis of control paths associated with a new pair of flow valves can be performed based on three different configurations (associated with the variable Conf in Line 16), as shown in Fig. 8(b-d). The first (2-Pin) configuration is applied when a new primary pin is needed. This configuration is adopted either at the first iteration (Line 12-14) or when there is a need for a new primary pin in order to prevent violation of the screening-latency constraint (Lines 23-24). In addition, a new demultiplexing pin is also needed. For example, in Fig. 8(b), a new primary pin C_1^a is connected to $\{V_1, V_2\}$ using a control valve l_1 that belongs to Level 1. Since l_1 is the first valve to be used in Level 1, a new demultiplexing pin C_1^g is connected.

The second and third configurations address the case where a new primary pin is not needed. These configurations, however, differ in how control valves are selected, and therefore, whether a demultiplexing pin is needed. To demonstrate the difference between the two configurations, we map the control valves and their connectivity into a binary tree, referred to as control-valves (CV) tree. Each node represents a connected control valve, and the root represents

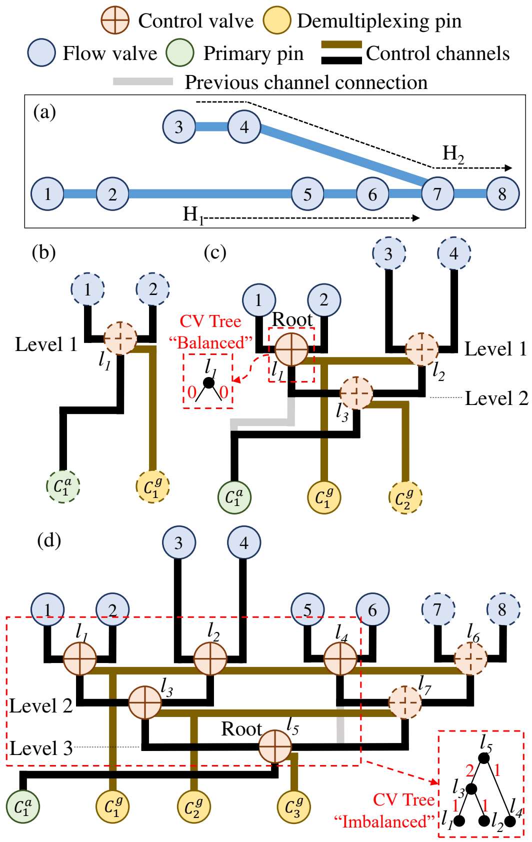

Fig. 8: Multiplexed control configurations used to connect a pair of flow valves. (a) An 8-valve RFB with two control sequences. (b) 2-Pin configuration. (c) 1-Pin configuration. (d) 0-Pin configuration.

the control valve at the highest level that is directly connected to the primary pin; see Fig. 8(c-d). Each node has two interfaces (left and right) which are connected to two sub-trees. Each interface is characterized by the maximum height of the associated sub-tree. A CV tree is balanced only if the interfaces of every node in the tree exhibit equal heights.

The CV tree evolves at every iteration whenever a new pair of flow valves is selected; the tree evolves in a bottom-up fashion; the root of the tree may change when new nodes are added. The second (1-Pin) configuration is selected if the CV tree is currently balanced. Fig. 8(c) shows an example for 1-Pin. Before the multiplexed-control scheme is expanded for connecting the pair $\{V_3, V_4\}$, we observe that the CV tree contains only one node l_1 and it is balanced. For connecting $\{V_3, V_4\}$ to the same primary pin C_1^a, two new control valves $\{l_2, l_3\}$ are connected. Since l_3 is the first valve to be connected at Level 2, a new demultiplexing pin C_2^g is selected and l_3 becomes the new root.

On the other hand, the third (0-Pin) configuration is applied if either the CV tree or any sub-tree within the CV tree is imbalanced. For example, in Fig. 8(d), we investigate the multiplexed-control paths when a new pair $\{V_7, V_8\}$ is selected. The CV tree is imbalanced, because the difference between the heights of l_5 interfaces is 1; this difference is called the imbalance factor. Valve l_5 is therefore called the source of imbalance and its level (Level 3) is called the level of imbalance. To connect $\{V_7, V_8\}$ to the primary pin C_1^a, two control valves $\{l_6, l_7\}$ are connected. The valve l_6 connects the pair $\{V_7, V_8\}$ and therefore it is located at Level 1. However, l_7 is located at Level z which is determined as follows: z = level of imbalance − imbalance factor. The control valve l_7 is used to connect the source of imbalance l_5, the new control valve L_6, and the previously connected valve l_4 . This configuration does not allocate new control pins. By using this balancing scheme, we reduce the height of the tree and hence decrease the number of demultiplexing pins.

The worst-case time complexity of this algorithm is $O(X \cdot |\mathcal{L}| \cdot A \cdot G)$.

V. Experimental Results

We implemented Sortex in a software simulation environment. All evaluations were carried out using a 3.4 GHz Intel i7 CPU with 12 GB RAM. The architecture of single-cell screening biochip (Section II-B) was used as a benchmark, and two biochip configurations were adopted; see Table I (# FV: number of flow valves; Max. # CV: maximum number of control valves; Max. # CP: maximum number of control pins). During simulation, we set an upper bound on the number of control entities (i.e., control pins and valves); locations of these entities were specified in advance.

We evaluate the performance of Sortex using two metrics: (1) the worst-case screening latency, measured in seconds (s); (2) the number of used control pins. In all evaluations, we consider air as a channel medium ($\mu = 1.983 * 10^{-5} Pa \cdot s$), and the channel dilatability is set to $\gamma = 10^{-5} Pa^{-1}$ (PDMS).

A. Comparison with Direct-Addressing

We compare Sortex with the direct-addressing (**DA**) method, in which every valve is addressed by a dedicated control pin. We evaluate **DA** and Sortex in terms of the screening latency and we study the convergence of latency for Sortex by varying the number of control pins. Fig. 9(a-b) show the screening-latency results for CF$_1$ and CF$_2$, respectively.

Based on Fig. 9, we observe that the latency decreases when the number of control pins is increased. With more control pins, the complexity of multiplexed control decreases, therefore the latency decreases. Moreover, the latency for Sortex converges to that for **DA**, despite the significant gap in the number of control pins between these methods.

B. Impact of Chip Parameters and Constraints

We next evaluate the impact of chip parameters and design constraints on screening performance. We focus only on the channel-width parameter and the worst-case latency.

TABLE I: Biochip configurations used in evaluation

Conf.	K, W, N	# FV	Max. # CV	Max. # CP
CF$_1$	4,2,1	28	140	28
CF$_2$	8,4,1	80	400	80

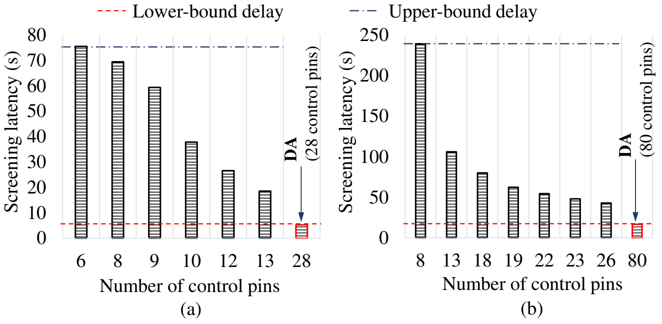

Fig. 9: Comparison between Sortex and **DA**: (a) screening latency (τ) for CF$_1$, (b) τ for CF$_2$.

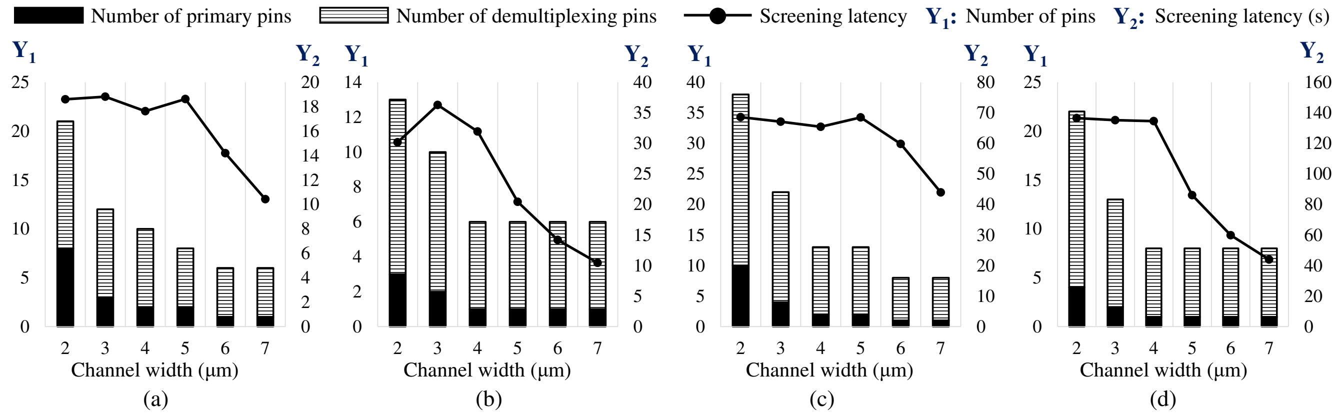

Fig. 10: Impact of channel width and latency constraint η on performance: (a) CF_1, η = 19 s; (b) CF_1, η = 38 s; (c) CF_2, η = 70 s; (d) CF_2, η = 140 s.

For each biochip configuration, we carried out two sets of synthesis simulations. In each set, we consider a specific latency constraint and report the number of control pins and the worst-case screening latency while varying the channel width. The latency constraints considered for CF_1 are 19 s and 38 s, and their associated results are shown in Fig. 10(a) and Fig. 10(b), respectively. Also, the latency constraints for CF_2 are 70 s and 140 s, and their associated results are depicted in Fig. 10(c) and Fig. 10(d), respectively.

We first investigate the number of control pins in Fig. 10. In all four cases, we observe that fewer control pins are needed (to satisfy the latency constraint) when the channel width is increased. This result is intuitive because using a wider channel causes the Elmore delay for fluid transport to be minimized, thus reducing the number divide-and-conquer procedures in Sortex (Section IV-B).

Second, we investigate the screening latency reported in Fig. 10. We observe that in Fig. 10(b) the latency first increases when the channel width is increased from 2 μm to 3 μm. Since the increase in the channel width leads to pin-count reduction, the impact of pin-count reduction on increasing the latency surpasses the impact of increasing the channel width on reducing the latency. We also observe that the latency is decreased in Fig. 10(b) when the channel width is larger than 3 μm as no further pin-count reduction can be achieved, i.e., the minimum number of control pins is reached ($\lceil \log_2 X \rceil + 1$). The same argument also applies to all other cases in Fig. 10. These results show the impact of the channel width, particularly narrow channels, on the screening performance.

We finally investigate the impact of changing the latency constraint on the number of control pins. As expected, by relaxing the latency constraint (i.e., increasing its value), the number of control pins is reduced. For example, by comparing Fig. 10(c) with Fig. 10(d), the number of control pins is decreased from 38 to 22 for a design with a 2 μm-channel width. While pin-constrained design has been studied earlier [6], [7], the trade-off analysis presented in this section cannot be applied to these prior methods; therefore a meaningful comparison is not applicable in this case.

VI. Conclusion

We have introduced a timing-driven design method for a pin-constrained RFB that performs single-cell screening. The proposed design synthesizes multiplexed control and employs a delay model of pressure-driven transport to satisfy a screening-delay constraint. The proposed method has been evaluated based on the screening delay and the pin count.

Acknowledgment

M. Ibrahim and K. Chakrabarty were supported in part by the US National Science Foundation under grants CCF-1702596 and CCF-1135853. In addition, K. Chakrabarty was supported by the Technische Universität München – Institute for Advanced Study, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement N° 291763.

References

[1] J. Kim *et al.*, "Pneumatically actuated microvalve circuits for programmable automation of chemical and biochemical analysis," *Lab on a Chip*, vol. 16, no. 5, pp. 812–819, 2016.

[2] J. Baret *et al.*, "Fluorescence-activated droplet sorting (FADS): efficient microfluidic cell sorting based on enzymatic activity," *Lab on a Chip*, vol. 9, no. 13, pp. 1850–1858, 2009.

[3] R. Zilionis *et al.*, "Single-cell barcoding and sequencing using droplet microfluidics," *Nature Protocols*, vol. 12, no. 1, pp. 44–73, 2017.

[4] I. Araci *et al.*, "Microfluidic very large scale integration (mVLSI) with integrated micromechanical valves," *Lab on a Chip*, vol. 12, no. 16, pp. 2803–2806, 2012.

[5] W. Minhass *et al.*, "Architectural synthesis of flow-based microfluidic large-scale integration biochips," *CASES*, 2012.

[6] K. Hu *et al.*, "Control-layer routing and control-pin minimization for flow-based microfluidic biochips," *IEEE Trans. TCAD*, vol. 36, no. 1, pp. 55–68, 2017.

[7] T. Tseng *et al.*, "Columba: co-layout synthesis for continuous-flow microfluidic biochips," in *DAC*, 2016.

[8] M. Ibrahim *et al.*, "Cosyn: Efficient single-cell analysis using a hybrid microfluidic platform," in *DATE*, 2017.

[9] D. Kim *et al.*, "A method for dynamic system characterization using hydraulic series resistance," *Lab on a Chip*, vol. 6, no. 5, pp. 639–644, 2006.

[10] B. Kirby, *Micro-and Nanoscale Fluid Mechanics: Transport in Microfluidic Devices*. Cambridge University Press, 2010.

[11] "COMSOL Multiphysics Modeling Software," http://www.comsol.com/.

[12] W. Grover *et al.*, "Development and multiplexed control of latching pneumatic valves using microfluidic logical structures," *Lab on a Chip*, vol. 6, no. 5, pp. 623–631, 2006.

[13] N. Amin *et al.*, "Computer-aided design for microfluidic chips based on multilayer soft lithography," in *ICCD*, 2009.

[14] W. Minhass *et al.*, "Control synthesis for the flow-based microfluidic large-scale integration biochips," *ASPDAC*, 2013.

[15] H. Yao *et al.*, "Integrated flow-control codesign methodology for flow-based microfluidic biochips," *IEEE Des. Test.*, 2015.

[16] J. Cong *et al.*, "Performance optimization of VLSI interconnect layout," *Integ., VLSI J.*, 1996.

[17] F. Perdigones *et al.*, "Correspondence between electronics and fluids in MEMS: Designing microfluidic systems using electronics," *IEEE Trans. Ind. Electron.*, vol. 8, no. 4, pp. 6–17, 2014.

[18] P. Tabeling, *Introduction to Microfluidics*. Oxford University Press, 2005.