Design Automation and Test Techniques for Microfluidic Biochips[1]

William L. Hwang

Advisors: Professor Krishnendu Chakrabarty, Professor Richard Fair
Collaborating Mentor: Dr. Fei Su
Department of Electrical & Computer Engineering
Pratt School of Engineering
Duke University
Spring 2006

Submitted in partial fulfillment of the requirement for Graduation with Departmental Distinction for the degree of Bachelor of Science in Engineering (BSE) in the Department of Electrical and Computer Engineering

Abstract

Microfluidics-based biochips, also referred to as lab-on-a-chip (LoC), are devices that integrate fluid-handling functions such as sample preparation, analysis, separation, and detection. This emerging technology combines electronics with biology to open new application areas including point-of-care diagnosis, on-chip DNA analysis, and automated drug discovery. As digital microfluidic biochips are developed for high throughput and concurrent assays, biochip-specific design automation and testing techniques are needed to handle increased complexity. In this paper, we address three important problems in the automation of design and testing of digital microfluidic biochips. First, we focus on the droplet routing problem, which is a key issue in biochip physical design automation. We develop the first systematic droplet routing method that can be integrated with biochip synthesis [1]. The proposed approach attempts to minimize the number of cells used for droplet routing, while satisfying constraints imposed by throughput considerations and fluidic properties. Next, we describe recent experiments that reveal the inadequacy of testing based on Hamiltonian paths in a graph model of the microfluidic array and introduce a novel testing methodology for localizing faults. We show previous approaches of mapping the droplet flow path problem to that of finding a Hamiltonian path in a graph model of the array is not sufficient to detect electrode-short and fluidic-open faults that affect two adjacent electrodes. To detect these edge-dependent defects, we developed testing methods based on adaptations of the Euler circuit and Euler path using a probabilistic-modified Fleury's algorithm to find efficient ways to detect and localize faults by traversing all edges of an undirected graph exactly once [2]. Finally, we propose a design automation method for pin-constrained biochips. In contrast to the direct-addressing scheme that has been studied thus far in the literature, we assign a reduced number of independent control pins to a large number of electrodes in the LoC, thereby reducing design complexity and product cost. We develop a virtual partitioning scheme that can prevent multi-droplet interference and maintain throughput for most practical assays on pin-constrained biochips [3]. We demonstrate the proposed method on a set of multiplexed bioassays.

[1] This research was supported by the National Science Foundation under grant number IIS-0312352 and a Research Experiences for Undergraduates (REU) grant.

1. Introduction

Microfluidic biochips are a promising class of mixed-technology devices with the potential to revolutionize biosensing, clinical diagnostics and drug discovery due to their small size and sample volumes, lower cost, and higher sensitivity compared to conventional biochemical laboratory methods. Microfluidic biochips are based on the precise control of microliter and nanoliter volumes of liquids, and they integrate various fluid-handling functions, such as sample preparation, analysis, separation, and detection [4,5,6]. This emerging technology combines electronics with biology to open new areas of research and field applications, e.g., point-of-care diagnosis, on-chip DNA analysis, and drug screening [5,6].

Currently, most commercially-available microfluidic devices rely on continuous-flow in etched microchannels [4,7,8]. An alternative paradigm is systems that utilize discrete droplets with microliter and nanoliter volumes. The droplets in such systems are actuated using electrical means, thereby obviating the need for cumbersome micropumps and microvalves. Droplet-based systems also offer dynamic reconfigurability and architectural scalability [9,10]. Droplet motion in such devices is typically controlled by a system clock, which is similar in operation to a digital microprocessor. Thus, this novel technology is referred to as "digital microfluidics".

Microfluidic processing is performed on unit-sized fluid droplets that are transported, stored, mixed, reacted, or analyzed in a discrete manner using a standard set of basic instructions. These fundamental instructions can be combined into hierarchical design structures that enable the step-by-step realization of complex procedures, such as chemical syntheses and biological assays. Unlike continuous-flow microfluidics, digital microfluidics has the advantage of significantly smaller volumes and higher levels of automation. As a result, an immense range of laboratory-established chemical protocols can be seamlessly transferred to a nanoliter droplet format.

The remainder of the paper is organized as follows. In Section 2, we present an overview of digital microfluidic biochips. In Sections 3-5, we present three critical problems in designing and testing digital microfluidic biochips. Finally, conclusions are discussed in Section 6.

2. Digital Microfluidic Biochips

A typical digital microfluidic platform consists of two parallel glass plates (Figure 1). The droplets containing biochemical samples and the filler medium (i.e., silicone oil) are sandwiched between the plates with the droplets traveling within the filler medium.

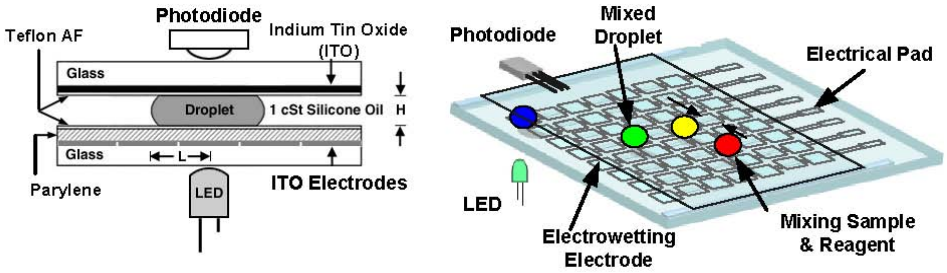


Figure 1. Schematics of a digital microfluidic biochip.

The bottom plate contains a patterned array of independently addressable control electrodes, and the top plate is coated with a continuous ground electrode. Each droplet rests on a hydrophobic surface over an electrode that controls its wettability through the application of an electric field. The actuation mechanism is based on the principle of electrowetting, a

phenomenon that involves the direct electrical control of droplet surface tension (Figure 2). Interfacial tension gradients can be created to move the droplets to the charged electrode by switching the voltage applied across adjacent electrodes. This mechanism enables droplets to be moved to any location on a two-dimensional array. The seminal invention of electrowetting-based actuation of liquid droplets for applications in digital microfluidics occurred in 2000 at Duke University [9]. Many applications have since been demonstrated, including clinical diagnostics on human physiological fluids and the polymerase chain reaction (PCR) for massively amplifying DNA samples, and considerable efforts are now focused on development of an efficient and rapid DNA pyrosequencing platform.

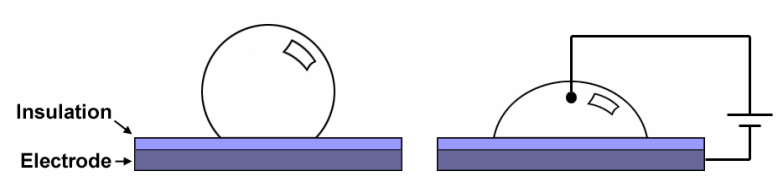


Figure 2. (Left) *No Potential*: A droplet on a hydrophobic surface has a large contact angle. (Right) *Applied Potential*: The droplet's surface energy increases, which results in a reduced contact angle. The droplet now wets the surface.

Using a two-dimensional digital microfluidic platform, many basic microfluidic operations for different bioassays can be performed by varying the patterns of control voltage activation, such as sample introduction (*dispense*), movement (*transport*), temporary preservation (*store*), sample dilution with buffer (*dilute*), and the mixing of different samples (*mix)*. A droplet can be held in a "virtual chamber" by applying an insulating voltage around it. On the other hand, mixing can be performed by routing two droplets to the same location and then turning them about some pivot points [11]. Since the liquid is manipulated by software programming, these operations can be performed anywhere on the array. This is in contrast to continuous-flow systems, where the fluidic operations must be executed in specific micromixers or microchambers fixed on a substrate. This flexibility is referred to as the reconfigurability of digital microfluidic biochips. The configurations of the microfluidic array and droplet routes can be programmed into a microcontroller that controls the voltages of electrodes in the array as function of time. In addition to the reconfigurable microfluidic array, optical detectors such as LEDs and photodiodes are also often integrated into digital microfluidic arrays to monitor colorimetric bioassays [6].

Another advantage of digital microfluidic biochips is in the flexibility of its designs. As in digital circuit design, a module-based method can be applied to the design of digital microfluidic biochips. A microfluidic module library, analogous to a standard/custom cell library used in cell-based VLSI design, includes different microfluidic functional modules, such as mixers and storage units. We can map the microfluidic assay operations to available microfluidic modules, and then use architectural-level synthesis techniques to determine a schedule of sets of bioassays subject to precedence constraints imposed by the corresponding assay protocols [12]. The locations of the modules on the microfluidic array are then determined by placement algorithms [13]. Therefore, these modules can be dynamically formed by activating the corresponding control electrodes during run-time. In this sense, the microfluidic modules used during assay operation can be viewed as virtual devices.

Printed Circuit Board (PCB) technology has recently been proposed as a means to rapidly prototype and inexpensively mass-fabricate digital microfluidic biochips. Using a copper layer

for the electrodes, solder mask as the insulator, and a Teflon AF coating for hydrophobicity, the microfluidic array platform can be fabricated by using an existing PCB manufacturing technology without any special modifications to the commercial process [14,15]. We further envision that a disposable PCB-based microfluidic biochip can be easily plugged into a controller circuit board that is programmed and powered via a standard USB port (Figure 3). A challenge to overcome in making these devices marketable is the high manufacturing cost. Methods to reduce the number of independent control pins will decrease the manufacturing cost of commercial digital microfluidic systems (Section 5).

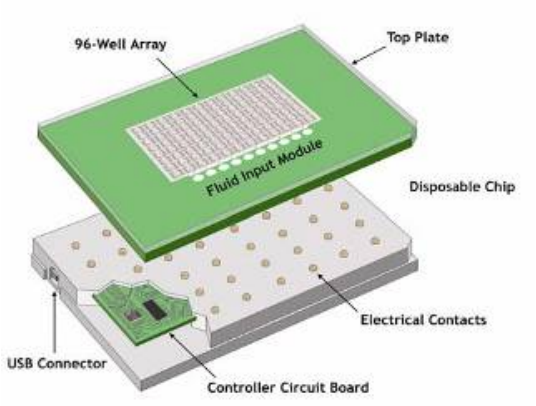


Figure 3. The concept of a commercial, PCB-based disposable microfluidic biochip.

3. Fluidic Constraints and Droplet Routing in Physical Design of Digital Microfluidic Biochips[2]

The level of system integration and design complexity of digital microfluidic biochips is expected to increase substantially as more high-throughput bioassays are executed concurrently on the same platform [16]. Current full-custom design methods are inadequate for digital microfluidic biochips that execute a set of simultaneous sensing tasks and require a high degree of run-time flexibility.

While research on device-level physical modeling and simulation of single microfluidic components has gained momentum in recent years [17,18], less attention has been devoted thus far to system-level design automation tools. However, in order to properly harness the potential offered by digital microfluidics, we need to deliver to biochip designers the same level of system-level CAD support that is now commonplace in the IC industry. Thus, biochip-specific design automation techniques are especially important for this emerging marketplace, especially in the area of computer-aided-design (CAD) tools for automated design and prototyping.[3]

Analogous to classical VLSI synthesis, a top-down system-level design automation approach can be used to relieve biochip users from the burden of manual optimization of assays and time-consuming hardware design. We divide the synthesis procedure for a digital microfluidic biochip into two major phases: architectural-level synthesis and physical design. A behavioral model for a set of bioassays is first obtained from laboratory protocols. Architectural-

[2] F. Su, W.L. Hwang, and K. Chakrabarty, "Droplet routing in the synthesis of digital microfluidic biochips," *Proc. Design, Automation, and Test in Europe Conference,* 2006.

[3] The 2003 International Technology Roadmap for Semiconductors (ITRS) identifies the integration of electrochemical and electro-biological techniques as one of the system-level design challenges beginning 2009, when feature sizes shrink below 50 nm [9].

level synthesis is then used to generate a macroscopic structure of the biochip. Next, physical design creates the final layout of the biochip, consisting of the placement of microfluidic modules such as mixers and storage units, the routes that droplets take between different modules, and other geometrical details [13].

We develop the first systematic routing method for digital microfluidic biochips; our approach attempts to minimize the number of cells used for droplet routing, while satisfying constraints imposed by performance goals and fluidic properties. We focus on droplet routing between chip modules and between modules and on-chip I/O reservoirs—an important problem in biochip physical design. The dynamic reconfigurability inherent in digital microfluidics allows different droplet routes to share cells on the microfluidic array during different time intervals. In this sense, the routes in microfluidic biochips can be viewed as *virtual routes*, which make droplet routing different from the classical VLSI wire routing problem.

Wire routing is a well-studied problem in VLSI design. Harnessing the analogy between digital microfluidics and digital electronics, many classical VLSI routing techniques can be leveraged for the droplet routing problem [19,20,21,22]. However, there exist some important differences. For example, whereas electrical nets must not be short-circuited in VLSI routing, i.e., they cannot intersect each other, different droplet routes can be overlapped on some locations as long as they satisfy certain constraints to prevent undesirable droplet mixing. Consequentially, capacity constraints that result from fixed routing regions in VLSI design are not as important in droplet routing.

A channel routing problem for continuous-flow microfluidic biochips has been investigated in [23]. Since these biochips are fabricated in a single layer of glass, silicon or plastic, all microchannels must be routed in a planar fashion. Moreover, these microfabricated channels must not intersect. Thus, this routing problem is similar to the classical single-layer VLSI routing problem.

An approach for coordinating droplets in digital microfluidic biochips has been presented in [24]. By viewing the microfluidic array as a network, the authors reduced the droplet path-planning problem to a network flow problem. Since droplet motion is limited to the permanent "streets", this approach does not exploit some of the important advantages of digital microfluidics, such as dynamic reconfigurability.

3.1. Droplet Routing Problem Formulation

The main objective in routing is to find droplet routes with minimum lengths, where route length is measured by the number of cells in the path from the starting point to the end point (i.e., Manhattan path length). As discussed in [13], the microfluidic array consists of primary cells that are used in assay operations, and spare cells that can replace faulty primary cells to achieve fault tolerance. Thus, for a microfluidic array of fixed size, minimum-length droplet routes lead to minimization of the total number of cells used in droplet routing, thus freeing up more spare cells for fault tolerance. This is especially important for safety-critical systems, an important application area for digital microfluidic biochips.

As in the case of electronic circuits, the fluidic ports on the boundary of microfluidic modules are referred to as *pins*, and we assume that assignment of pins has been completed in the module placement phase. Similarly, we refer to the droplet routes between pins of different modules or on-chip reservoirs as *nets*. Thus, a fluidic route on which a single droplet is transported between two terminals can easily be modeled as a 2-pin net. We also often have a need to move two droplets from different terminals to one common mixer module. To allow

droplet mixing simultaneously during their transport, which is preferable for efficient assay operations [6], we need to model such fluidic routes as 3-pin nets, instead of two individual 2-pin nets.

3.2. Fluidic Constraints

During droplet routing, a minimum spacing between droplets must be maintained to prevent accidental mixing, except for the case when droplet merging is desired (e.g., in 3-pin nets). We view the microfluidic modules placed on the array as obstacles in droplet routing. In order to avoid conflicts between droplet routes and assay operations, a segregation region is added to wrap around the functional region of microfluidic modules. In this way, droplet routing can easily be isolated from active microfluidic modules.

For multiple droplet routes that may intersect or overlap with each other, static and dynamic fluidic constraint rules must be introduced to avoid undesirable behavior. Without loss of generality, we refer to two arbitrary droplets as D_i and D_j. First, to avoid mixing, we assume that their initial locations at time slot t are not directly adjacent or diagonally adjacent to each other. Let us represent the microfluidic array by two-dimensional Cartesian coordinates (X, Y), and let $X_i(t)$ and $Y_i(t)$ denote the location of D_i at time t. To keep the droplets from unintended mixing, we must ensure that either $|X_i(t) - X_j(t)| \geq 2$ or $|Y_i(t) - Y_j(t)| \geq 2$. Similarly, to select the admissible, non-mixing locations of the droplets at the next time slot t+1, we find that the following fluidic constraint rules need to be satisfied:

Rule 1: $|X_i(t+1) - X_j(t+1)| \geq 2$ or $|Y_i(t+1) - Y_j(t+1)| \geq 2$, i.e., their new locations are not adjacent to each other;

Rule 2: $|X_i(t+1) - X_j(t)| \geq 2$ or $|Y_i(t+1) - Y_j(t)| \geq 2$, i.e., the activated cell for droplet D_i at time t+1 cannot be adjacent to droplet D_j. Otherwise, there is more than one activated neighboring cell for D_j, which may lead to errant fluidic operation; and

Rule 3: $|X_i(t) - X_j(t+1)| \geq 2$ or $|Y_i(t) - Y_j(t+1)| \geq 2$.

Note that Rule 1 can be considered a static fluidic constraint, whereas Rules 2 and 3 are dynamic fluidic constraints. We verified these fluidic constraint rules through a set of laboratory experiments involving concurrent movement of two 2 μL droplets enveloped with 4 μL of 1 cSt silicone oil on an open (i.e., no top plate) 18433G PCB biochip. Electrodes 1 through 8 of this chip are labeled in Figure 4a. The experimental setup is shown in Figure 4b. Concurrent movement was established through the simultaneous activation of each electrode relevant to droplet motion.

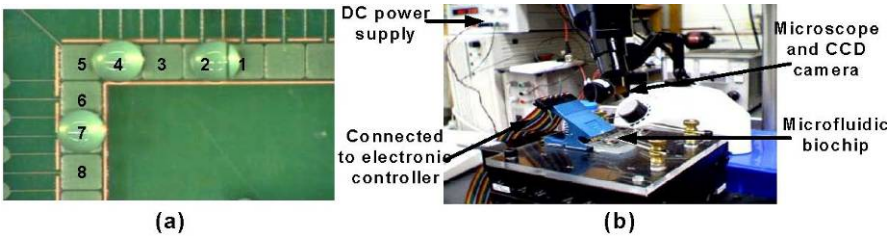


Figure 4. (a) Section of a PCB digital microfluidic biochip used for fluidic constraint experiments; (b) Experimental setup.

Rule 1 states that the new positions of two droplets at time t+1 must be no closer than two electrodes in each dimension. In Figure 5, droplets D_i and D_j were initially actuated on

electrodes 1 and 4 respectively, and then concurrently moved to electrodes 2 and 3. The new locations of the two droplets are adjacent and therefore merging occurred to minimize surface energy.

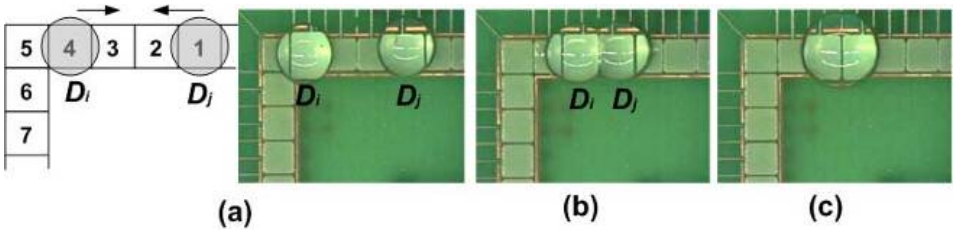


Figure 5. Experiments demonstrating that two aligned droplets moving towards each other must be separated by more than two electrodes to prevent merging (Rule 1) **(a)** Droplets begin on electrodes 1 and 4 **(b)** Electrodes 2 and 3 are activated and 1 and 4 deactivated; droplets move equally to merge **(c)** Merged droplet centered over electrodes 2 and 3.

The first dynamic fluidic constraint (Rule 2) is demonstrated in Figure 6. Droplets D_i and D_j were initially situated on non-adjacent electrodes 4 and 2, respectively. They were then simultaneously transported to electrodes 3 and 1, respectively. Although static constraints are satisfied, there were two activated neighboring cells for D_j. Predictably, it was observed that droplet D_j did not move off of electrode 2 since similar reductions of interfacial energy occurred on both sides of the droplet. On the other hand, D_i only had its interfacial energy reduced on the side it overlaps with electrode 3, and it moved to electrode 3 when it is activated concurrent with deactivation of electrode 4. With D_i and D_j now adjacent, the droplets merge.

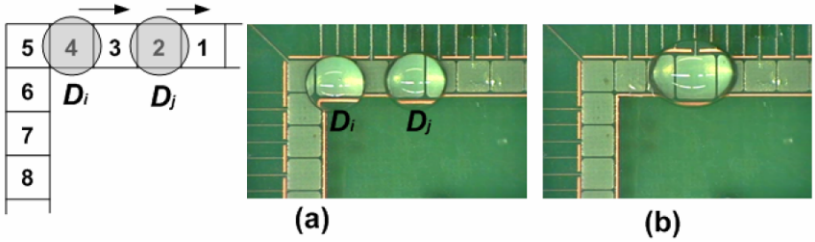


Figure 6. Experiments demonstrating that two aligned droplets moving in the same direction must be separated by more than one electrode to prevent merging. **(a)** Placement of droplets on electrodes 2 and 4 **(b)** Electrodes 1 and 3 are activated; *Dj* does not move appreciably while *Di* moves towards electrode 3 **(c)** Merged droplet regains spherical shape.

Furthermore, note in Figure 6a that the placement of droplet 2 was such that it overlaps much more with electrode 1 than 3. Even with such a large imbalance in overlap, if there are two simultaneously activated electrodes adjacent to the droplet, the droplet still does not show significant preference to move towards the electrode with which it has greater overlap.

The second dynamic fluidic constraint (Rule 3) can be shown with the experimental setup displayed in Figure 7. The two droplets now move in orthogonal directions. As shown in Figure 7a, droplets D_i and D_j were initially placed on electrodes 4 and 7 respectively. Electrodes 3 and 6 were then activated concurrently to move D_i and D_j rightward and upward, respectively. Merging occurs because D_i is adjacent to two activated cells (i.e., directly adjacent to 3 and diagonally adjacent to 6) and therefore does not move as predictably as D_j.

The purpose of the above experiments was to demonstrate that adherence to Rule 1 is not sufficient to prevent merging. Both Rule 2 and Rule 3 must also be satisfied during droplet routing. Moreover, these fluidic constraint rules can provide guidelines for modifying droplet motion (e.g., force some droplets to remain stationary in a given time-slot) to avoid constraint

violation if necessary: the details of such a strategy are discussed in Section 3.5.3.

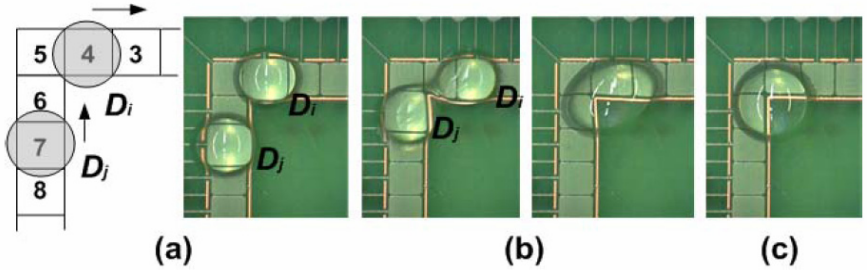


Figure 7. Experiments demonstrating that two orthogonal droplets moving in the same direction must be separated by more than two electrodes to prevent merging. **(a)** Placement of droplets on electrodes 4 and 7 **(b)** Electrodes 3 and 6 are activated; *Di* does not move appreciably while *Dj* moves towards electrode 6 **(c)** Merged droplet regains spherical shape.

3.3. Timing Constraints

Another important constraint in droplet routing is given by an upper limit on droplet transportation time. This constraint arises from a critical assumption made in the architectural-level synthesis of digital microfluidic biochips. It is assumed in [12] that since droplet movement on a microfluidic array is very fast compared to assay operations (e.g., mixing, dilution, and optical detection), we can ignore the droplet routing time for scheduling assay operations. This assumption has been validated by laboratory experiments for simple assays. For example, it has been reported that mixing in a 2×4-array mixer takes about 10 seconds, whereas it takes only 10 ms for a droplet to move across one cell (with 100 Hz clock frequency) [25].

To ensure that the above assumption is valid for complex sets of concurrent assays, we need to ensure that the delay for each droplet route does not exceed some maximum value, e.g., 10% of a time-slot used in scheduling. Otherwise, the schedule obtained from the synthesis procedure is no longer valid. This timing constraint is analogous to the interconnect delay constraints in VLSI routing that require each wire net (or critical path) to meet its timing budget. Note that since a droplet may be held at a location in some time slots during its route, the delay for a droplet route consists of the transport time combined with the idle time.

3.4. Problem Decomposition

Since digital microfluidic biochips can exploit the dynamic reconfigurability of the microfluidic array during run-time, a series of 2-D placement configurations in different time spans, instead of a single 2-D placement as in classical VLSI design, are obtained in the module placement phase [13]. In this way, we can decompose droplet routing into a series of sub-problems. In each sub-problem, the nets to be routed from the target module to the source module are determined first. Only the microfluidic modules that are active during this time interval are considered as obstacles in droplet routing. Next we attempt to find suitable routes for these nets. These sub-problems are addressed sequentially to obtain a complete solution for droplet routing.

3.5. Routing Method

This section presents the proposed droplet routing method. The quality of the solution obtained by this method is independent of the routing order of nets. The inputs to the algorithm are a list of nets to be routed in each sub-problem as well as constraints imposed by the designer.

The droplet routing algorithm consists of two basic stages. The first stage generates M alternative routes for each net, where $M \leq$ *total number of possible routes,* can be fine-tuned through experiments (e.g., we use $2 < M < 10$ in our work). The algorithm attempts to find the shortest routes for each net (2-pin net or 3-pin net). In addition, the obtained routes also need to pass the timing delay constraint check (TDCC) in this stage. Those that violate the timing constraint are pruned from the set of alternative routes. Note that if all the shortest routes for some net do not satisfy the timing constraint, placement refinement is required to increase routability.

The second stage of the routing algorithm first randomly selects a single route from the alternatives for each net. The set of randomly-selected routes for a given list of nets then go through the fluidic constraint rule check (FCRC). If necessary, some modifications are made to the routes to ensure that all fluidic constraints are satisfied. The delay for the corresponding net is updated, and then TDCC is performed again. The objective function for this set of routes is also obtained by calculating the number of cells used in routing. Through an appropriate number of random selection runs, a set of routes with the minimum cost function, subject to both timing and fluidic constraints, is finally selected to be the output of the routing algorithm. If no suitable solution for droplet routing is found, routability-oriented placement refinement is invoked again. Advantages of the routing methodology described above include the avoidance of the net-routing-order dependence problem, and the use of dynamic reconfigurability. Some key details of the algorithm are as follows.

3.5.1. Phase I: *M*-shortest Routes

In the first phase of the algorithm, M alternative routes for each net are generated. We modify the Lee algorithm, a popular technique used in grid routing [21,22], for the droplet routing problem in digital microfluidic biochips.

3.5.1.1. Two-pin nets

The shortest route problem for 2-pin nets is equivalent to the single-pair shortest path problem. An advantage of the Lee algorithm is that it is guaranteed to find the shortest path between two pins, which can be included among the M alternatives. We further modify the Lee algorithm to find the M shortest paths for each net, not all of which will have the overall shortest length.

3.5.1.2. Three-pin nets

We use 3-pin nets to model the routes along which two droplets are transported towards a microfluidic module (e.g., a mixer); the droplets can mix together during their transportation. The shortest-route problem for such nets is equivalent to the Steiner Minimum Tree (SMT) problem [21]. Since SMT is known to be NP-hard, a heuristic approach is needed. We can also modify the Lee algorithm to address this problem.

3.5.2. Phase II: Random Selection

In the second phase of the algorithm, a single route from the M_i alternatives for each net i is selected, where $i \in \{1,2,\ldots,N\}$ and N is the number of nets. Note that $M_i \leq M$ since some routes that violate the timing constraint have already been eliminated. A random selection approach is then used to select i_k for each net i, where i_k represents the k-th alternative route for

net i, and $k \in \{1, 2, \ldots, M_i\}$. A desirable feature of this random method is that it avoids the net-routing order dependence problem.

To evaluate the set of selected routes, we model the cost function as the total number of cells used in routing. It is represented by $C = \sum_{i=1}^{mxn} X_i$, where the binary variable X_i is 1 if cell i in an $m \times n$ array is used in the route; otherwise, it is 0.

Constraint checking also needs to be performed for each set of selected routes. If it fails TDCC or FCRC even after modifying droplet motion as discussed in Section 3.5.3, we assign a large penalty value P_t to this set of routes. Otherwise, we set $P_t = 0$ for those that satisfy all constraints. After an adequate number of random selection runs, we select the set of routes with the minimum cost value C and $P_t = 0$ as the output of the routing algorithm.

3.5.3. Fluidic Constraint Rule Check (FCRC) and Droplet Motion Modification

We note that in the first stage of the algorithm, M alternative routes for each net are obtained irrespective of the existence of other nets. However, since different net routes may share the same cells or they may be close to each other, we need to ensure that they do not violate the fluidic constraints stated in Section 3.2.

Assume that two droplet routes (i.e., R_i and R_j) have been obtained using the modified Lee algorithm. To adhere to fluidic constraint rules, we need to check the positions of these two droplets, D_i and D_j, in each time slot. Interestingly, even if a rule violation is found, we can still modify droplet motion (i.e., force a droplet to stay in the current cell instead of moving) to prevent the violation (Table I). If the stall modification fails (i.e., last row of Table I), the corresponding routing paths are deemed to be incompatible. We can further extend modification to the case of more than two droplet pathways.

Table I. Droplet motion modifications when fluidic constraints are violated.

Rule 1	Rule 2	Rule 3	Modification
Pass	Pass	Pass	Not required
Pass	Pass	Fail	D_j stays
Pass	Fail	Pass	D_i stays
Pass	Fail	Fail	N/A*
Fail	Pass	Pass	Droplet with the shorter pathlength stays
Fail	Pass	Fail	D_j stays
Fail	Fail	Pass	D_i stays
Fail	Fail	Fail	Fail

*N/A denotes that this case does not exist.

4. Defect Detection and Localization in Testing of Digital Microfluidic Biochips[4]

The current push towards shrinking processes and new materials, as well as the multiple underlying energy domains (e.g., electrical, mechanical, and fluidic) renders microfluidic biochips susceptible to manufacturing defects. Moreover, some defects are likely to be latent in the sense that they only manifest themselves during field operation of the biochips, and additional faults such as particle contamination may be introduced while operating in harsh

[4] F. Su, W.L. Hwang, A. Mukherjee, and K. Chakrabarty, "Defect-oriented testing and diagnosis of digital microfluidics-based biochips," *Proc. IEEE International Test Conference,* 2005.

environments. Consequently, it is paramount that robust off-line and on-line test techniques are available to detect manufacturing and field-operation induced faults and to reconfigure the chips by moving certain modules from defective sites to functional areas.

Faults in microfluidic biochips are classified as catastrophic or parametric. A catastrophic fault is one that leads to a cessation of droplet movement and complete malfunction of the system. Parametric faults are those that modify droplet behavior in a way that results in degradation of system performance, such as a decrease in the velocity of passing droplets, which may have an effect on scheduling/routing algorithms.

A previous test planning method for the detection of catastrophic faults was based on a graph model of the microfluidic array using Hamiltonian paths [26,27], which are paths that cover each *cell* of an array exactly once. By sending a test droplet along a predefined path that terminates at a capacitance detector, it is straightforward to determine whether there are catastrophic faults on the electrodes included in the test path. If the capacitance detector senses the droplet at the time point when the testing should be completed, then there are no catastrophic faults. However, if the droplet gets stuck somewhere along the path, the capacitance detector will sense that the test droplet has not arrived at its destination, and concludes that there must be a defect in the test path. Various systematic testing methods that involve numerous partially overlapping test paths can localize faults.

An efficient concurrent testing method that interleaves test application with a set of bioassays was developed [28] using Hamiltonian paths. However, we found that this work on testing digital microfluidic biochips is based on invalid assumptions regarding the impact of certain defects on droplet flow. Moreover, previous studies were restricted to investigating pass/fail tests, and not tests that *localize* faulty cells, which is important to facilitate reconfiguration of fluidic modules in order to bypass defective cells. We describe recent experiments that reveal the inadequacy of testing based on Hamiltonian paths in a graph model of the microfluidic array.

The underlying assumption in using Hamiltonian paths [27,28] for fault detection is that all faults are cell-based rather than edge-based. In other words, a faulty cell can be detected so long as a test droplet is actuated on it, regardless of the direction from which it arrived. For instance, to test for the electrode-open fault, it is sufficient to move a test droplet from any adjacent cell to the faulty cell. The droplet will always be stuck during its motion due to the inability to charge the control electrode. On the other hand, if we move a test droplet across the faulty cells affected by an electrode-short defect, the test droplet movement may be dependent on flow direction. To determine if this is indeed the case, we designed an experiment to evaluate the behavioral impacts of electrode-short faults.

Figure 8 illustrates the 2x4 glass microfluidic chip used to determine whether two valid Hamiltonian droplet flow paths may have differing abilities to detect certain electrode-short faults. The control electrodes in the bottom glass plate are formed by a 200 nm thick layer of chrome, which is further coated with a layer of Parylene C (800 nm) as a dielectric insulator. This microfluidic array uses a 1.0 mm electrode pitch size. A layer of optically transparent indium tin oxide (ITO) in the top glass plate is used as the continuous ground electrode. In addition, a 50 nm thick film of Teflon AF 1600 is added as the hydrophobic coating on both the top and the bottom plates. The 600 μm gap between the top and bottom plates is set using a glass spacer. To simulate electrode shorts, the two electrodes of interest were activated on the same clock cycle.

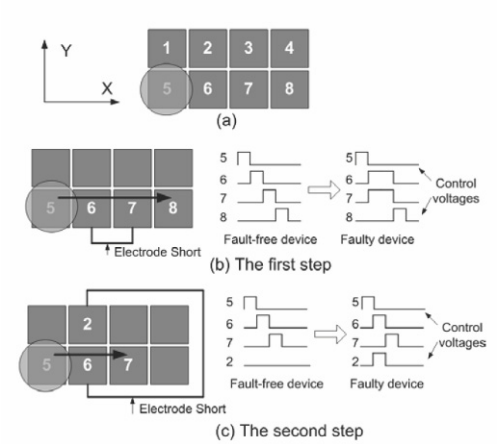


Figure 8. Experimental design to study droplet movement behavior in the presence of an electrode-short fault. Electrode shorts between two cells are simulated by activating both cells on the same clock cycle.

The DC actuation voltage was set to 50 V and a 1 μL droplet containing 0.1 M KCl was used and 1 cSt silicone oil served as the filler medium. First, a test droplet was transported across two shorted electrodes parallel to the droplet flow direction. The droplet had a tendency to align itself in between the two charged electrodes so as to maximize overlap area and thereby minimize the electrostatic energy that was stored between the droplet-electrode capacitive system. Thus, if a test droplet traverses the electrode short in the parallel direction, its progress will be halted and the fault will be detected, as shown in Figure 9.

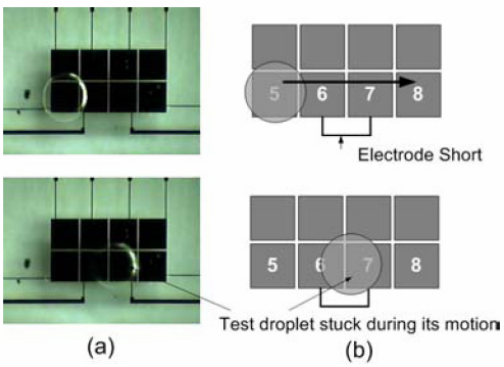


Figure 9. Experimental demonstration that a droplet becomes stuck between the two electrodes involved in an electrode short if the direction of droplet transport is aligned with the short.

On the other hand, if the test droplet traverses the electrode short orthogonally as shown in Figure 10, the test droplet can bypass the fault and continue on its intended path. For this simple 2x4 array, if the droplet finishes its Hamiltonian path by moving up to the top row and traveling to the left, it will not detect the fault on the second pass either. This "orthogonal" Hamiltonian path is not capable of detecting a short between electrodes 2 and 6.

These experimental results provide useful insights on how testing should be carried out for microfluidic arrays. Electrode short faults lead to an error only when the droplet flow path is aligned with the orientation of the electrode shorts. In addition to electrode shorts, there exist other physical defects that lead to similar erroneous behavior. For example, particle contamination between two adjacent cells also produces an error under specific droplet flow paths. In order to detect these defects, a test plan should guide the test droplet from a cell in the array to all its neighbors.

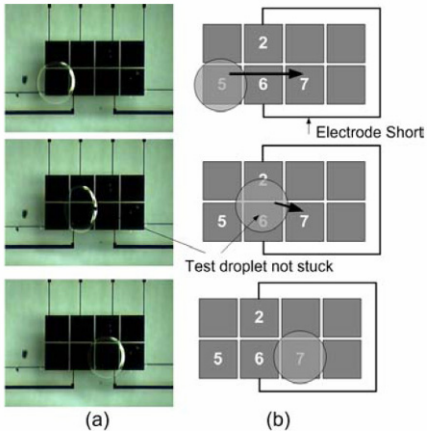


Figure 10. Experimental demonstration that a droplet bypasses the two electrodes involved in an electrode short if the direction of droplet transport is orthogonal to the short.

These experimental results highlight a major deficiency of prior work on the testing of microfluidic arrays [27,28]. The previous approaches map the droplet flow path problem to that of finding a Hamiltonian path in a graph model of the array. In other words, the test droplet is routed through the array such that it visits every cell exactly once. While this approach guarantees the detection of faults involving only one electrode or cell, it is not sufficient to detect electrode-short and fluidic-open faults that affect two adjacent electrodes. To detect these edge-dependent defects, off-line and on-line testing methods that can be implemented concurrently with bioassays have been developed.

First, we model the array of microfluidic cells using an undirected graph $G = (V, E)$ where the set of vertices V represents the set of microfluidic cells in the array, and each edge is an unordered pair of vertices. The edge $\{u,v\} \in E$ iff vertex u and vertex v represent two directly adjacent microfluidic cells. Figure 11(a) shows an undirected graph model for a 5×5 microfluidic array.

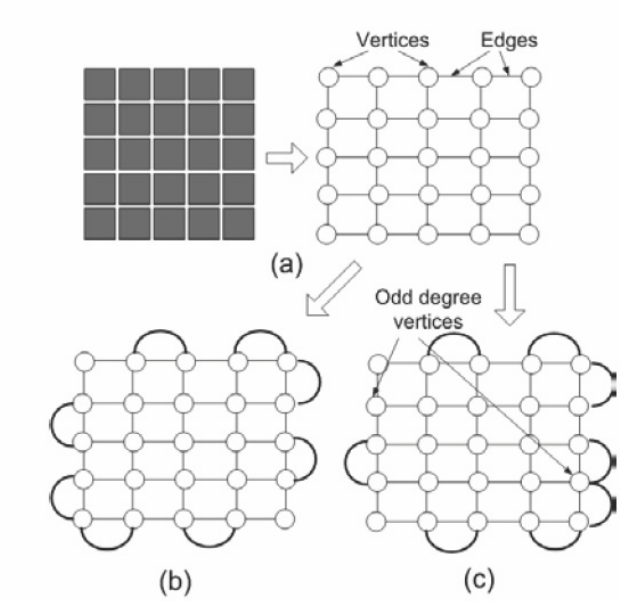


Figure 11. (a) Undirected graph model for a 5×5 microfluidic array; (b) eulerized graph containing an Euler circuit; (c) eulerized graph containing an Euler path.

An Euler path in a graph *G* is defined as a path that traverses all the edges of *G* exactly once [29]. Similarly, an Euler circuit is an Euler path that is also a cycle. We know from [29] that an undirected graph has an Euler circuit iff it is connected, and each vertex has even degree. Moreover, an undirected graph has an Euler path if it is connected and has exactly two vertices of odd degree. The Euler path must start at one of the odd-degree vertices and end at the other odd-degree vertex [29].

Euler's theorems give us the means for finding efficient ways in which to traverse all the edges of an undirected graph. However, we notice that a graph model of a microfluidic array usually has more than two vertices of odd degree. Thus we have to retrace some of the edges in order to traverse all edges at least once. To minimize the retracing, we can convert the vertices of odd degree to even degree by adding additional edges. The process of eliminating odd degree vertices by adding additional edges is called *eulerizing* the graph. There are two different ways for eulerizing the graph model of a microfluidic array, depending on whether an Euler circuit or an Euler path is desired. For example, as shown in Figure 6(b), there exists an Euler circuit in the eulerized graph model for a 5×5 microfluidic array since each vertex becomes to be even degree. On the other hand, another eulerized graph in Figure 6(c) contains an Euler path starting from one odd-degree vertex, e.g., cell (2,1) and ending at another odd-degree vertex, e.g., cell (4, 5).

Although both eulerizing methods can provide an edge tour as the feasible flow path of a test droplet, we use the Euler circuit method here. There are two main reasons for this choice. First, in the Euler path method we must use the node with odd degree as the starting or the ending point. Thus, to find an Euler path between another pair of cells, a different eulerized graph is required. In contrast, since any vertex can be used as the start and end point of an Euler circuit, we can locate the test droplet source/sink adjacent to any boundary cell using the same eulerized graph in the first method. Thus, this method is especially suitable when we try to determine the optimal location of droplet sources and sinks. Second, we are motivated by considerations of physical implementation. If we merge the test droplet source and sink, i.e., connect the electrode of the dispensing port to the capacitive detection circuit, it not only reduces the area overhead of the test hardware, but it can also conserve the liquid volume of on-chip reservoir by recycling test droplets. This reduces the cost of manual maintenance. This feature is especially desirable for in-field testing.

To find an Euler circuit in the eulerized graph model of an array, we employ the well-known Fleury's algorithm. Pseudocode for this algorithm is listed below. More details on our adaptation of Fleury's algorithm can be found in [2].

FLEURY'S ALGORITHM
1 Make sure the graph is connected and all vertices have even degree
2 Start at any vertex
3 Travel through an edge that is not visited if
    a) it is not a bridge for the part not visited, or
    b) there is no other alternative
4 Label the edges in the order in which they were visited
5 When there are no unvisited edges, an Euler circuit is found.

5. Design of Pin-Constrained Digital Microfluidic Biochips[5]

Various methods can be used to activate individual cells in a microfluidic array. In direct-addressing schemes, each cell of the patterned electrodes can be accessed directly and independently via a dedicated control pin. Most design and CAD research for digital microfluidic biochips has been focused on such directly-addressable chips [12,13,27]. This method is suitable for small/medium-scale microfluidic electrode arrays (i.e., with fewer than 10×10 electrodes). However, for emerging large-scale digital microfluidic biochips (i.e., with >100×100 electrodes), multi-layer electrical connection structures and complicated routing solutions are required due to the large number of independent control pins (e.g., 10^4 pins for a 100×100 array). Product cost, however, is a major marketability driver due to the disposable nature of most emerging devices. Hence, simpler routing solutions are necessary.

Recently, multi-layer printed circuit board (PCB) technologies have been used to fabricate electrical connections for digital microfluidic biochips [14]. However, multiple metal layers for PCB design may lead to reliability problems and also increase the fabrication cost. Thus, reducing the number of independent control pins is important for successful commercialization.

We propose a new design automation method for pin-constrained digital microfluidic biochips. In contrast to the direct-addressing scheme, a small number of independent control pins are assigned to a large number of electrodes, thereby dramatically reducing the input bandwidth between the electronic controller and the microfluidic array while minimizing any decrease in performance. We evaluate the proposed method by applying it to an array used for multiplexed bioassays.

A problem related to pin-constrained design of digital microfluidic biochips has been analyzed in [6]. In order to minimize the number of electrical contacts, the fabricated electrowetting-based biochip uses a multi-phase bus for the fluidic pathways. In an n-phase bus, every nth electrode is electrically connected. Thus, only n control pins are needed for a transport bus, irrespective of the number of electrodes that it contains. Although the multi-phase bus method is useful for reducing the number of control pins, it is only applicable to a one-dimensional (linear) array.

In other recent work, [30] presents a cross-reference driving scheme for digital microfluidic biochips, which allows control of an $N \times M$ grid array with only $N + M$ control pins. In this method, electrode rows are patterned on both the top and bottom plates, and placed orthogonally. In order to drive a droplet along the X-direction, electrode rows on the bottom plate serve as driving electrodes, while electrode rows on the top serve as reference ground electrodes [30]. The roles are reversed for movement along the Y-direction. This cross-reference method facilitates the reduction of control pins. However, it requires a special electrode structure (i.e., both top and bottom plates containing electrode rows), which results in increased manufacturing cost for disposable microfluidic chips. Moreover, this design is not suitable for high-throughput assays.

5.1. Pin Assignment Problem Formulation

In this section, we formulate the application-independent pin-constraint problem for a two-dimensional electrode array.

[5] W.L. Hwang, F. Su, and K. Chakrabarty, "Automated design of pin-constrained digital microfluidic arrays for lab-on-a-chip applications," accepted for publication in *Proc. IEEE/ACM Design Automation Conference,* 2006.

5.1.1. Minimum Number of Pins for a Single Droplet

Given a two-dimensional microfluidic chip, the problem of determining the minimum number of independent control pins, k, necessary to have full control of a single droplet can be reduced to the well-known graph-coloring problem [31]. Full control implies that a droplet can be moved to any cell on the array through an appropriate activation sequence.

While the problem of finding the chromatic number of a graph is NP-hard [32], it is trivial to observe that for rectangular arrays of size greater than 3 x 3, the largest number of directly adjacent neighbors to any cell is four. Hence, we set the number of independent control pins, $k \geq 5$, in order to guarantee that we can color each cell and all of its directly adjacent neighbors with different colors. A possible pin layout using 5 pins for a 4x4 array is shown as an example in Figure 12(a). If diagonally adjacent neighbors have an effect on a droplet in a particular cell, then we must account for eight neighbors, yielding the condition $k \geq 9$; one possible 4x4 layout is shown in Figure 12(b). Note that increases in array size do not change the required number of independent pins for full control of a single droplet.

1	2	3	5
3	5	4	1
4	1	2	3
2	3	5	4

(a)

1	2	3	8
8	7	6	5
5	4	9	1
1	2	3	8

(b)

Figure 12. (a) 5-pin layout, and (b) 9-pin layout for a 4x4 array (the entries in the array refer to pin numbers).

We use $k = 9$ in most of the discussion that follows even though we have experimentally verified that the diagonally adjacent neighbors have no effect on a droplet at a particular cell when the electrode size is sufficiently small. The choice of $k = 9$ is motivated by the observation that five independent pins are too few to reduce interference between multiple droplets to a manageable level.

5.1.2. Pin-Assignment Problem for Two Droplets

We first examine the interference problem for two droplets. For multiple droplets, the interference problem can be reduced to the two-droplet problem by examining all possible pairs of droplets. For the independent control of d droplets, all $\binom{d}{2}$ pairs must be checked for interference to determine if all droplets can safely perform their desired operations. In general, any sequence of movements for multiple droplets can occur in parallel, controlled by a clock. We analyze interference between two droplets for a single clock cycle, during which time a droplet can only move to an adjacent cell. Any path can be decomposed into unit movements, and we say that the paths of two droplets do not interfere only if all of their individual steps do not interfere.

In some situations, we would like both droplets to move to another cell in the next clock cycle. If this is not possible without interference, then a contingency plan is to have one droplet undergo a stall cycle (i.e., stay on its current cell). There are other possibilities such as an evasive move or backtracking to avoid interference, but these lead to more substantial changes in the scheduled droplet paths and are therefore not considered in this paper.

Let us denote two droplets by D_i and D_j, respectively with the position of droplet D_i at

time t given by $P_i(t)$. Let $N_i(t)$ be the set of directly adjacent neighbors of droplet D_i as a function of time. The operator $\bar{k}(\bullet)$ is the set of pins (no redundancies) that control the set of cells given by •. Then the problem of two droplets moving concurrently can be formally stated as:

- D_i moves from $P_i(t)$ to $P_i(t+1)$
- D_j moves from $P_j(t)$ to $P_j(t+1)$

To simplify the derivation of the interference constraints, we assume that droplets can only be affected by their directly adjacent neighbors. Although current electrode sizes (e.g., 1.5 mm x 1.5 mm) for rectangular PCB biochips are large enough for the diagonal adjacent effect to be significant, experimental work suggests that at smaller electrode sizes, the effect diminishes to the point of being negligible.

We are interested in the overlap of *pins* between sets of cells for the interference constraints, not the spatial locations of the cells. The latter are important for the fluidic constraints discussed in [1]. For the purpose of defining interference behavior, the system is completely determined by the positional states of the two droplets at times t and $t+1$. For droplet D_i, the positional states are characterized by the quartet ($P_i(t)$, $P_i(t+1)$, $N_i(t)$, $N_i(t+1)$) and for D_j, the quartet ($P_j(t)$, $P_j(t+1)$, $N_j(t)$, $N_j(t+1)$). We consider $\bar{k}(\bullet)$ of all unordered pairs involving the following sets: $P(t)$, $P_i(t+1)$, $N_i(t)$, $N_i(t+1)$, $P_j(t)$, $P_j(t+1)$, $N_j(t)$, $N_j(t+1)$. Since, there are $\binom{8}{2} = 28$, there are 28 such pairs that may need to be mutually exclusive to prevent interference between the droplets. Pairs of cell sets that must be mutually exclusive for non-interference must be contained in this pool of unordered pairs because these eight sets include all cells currently occupied by droplets and all of their neighbors. Twenty-two pairs can be quickly removed from consideration, leaving only six pairs of sets that need to be closely examined to determine if their mutual exclusion is required for non-interference. For simplicity, the pin operator $\bar{k}(\bullet)$ is left implicit in the following discussion. Mutual exclusion always refers to the *pins* controlling the cells, and not the cells themselves.

Pair 1 $\{P_i(t), N_j(t)\}$: if $N_j(t)$ contains a cell that shares the same pin as $P_i(t)$, then D_j may be between $P_j(t)$ and $P_j(t+1)$ at time t. If this is not the case, D_j may not be able to move to $P_j(t+1)$ because it will no longer overlap with $P_j(t+1)$.

Pair 2 $\{P_i(t+1), N_j(t)\}$: if $N_j(t)$ contains a cell that shares the same pin as $P_i(t+1)$, D_j will not move properly at time $t+1$ unless $P_j(t+1)$ is the cell in $N_j(t)$ that shares the same pin as $P_i(t+1)$.

Pair 3 $\{P_i(t+1), N_j(t+1)\}$: if $N_j(t+1)$ contains a cell that shares the same pin as $P_i(t+1)$, then D_j will drift after moving to $P_j(t+1)$ so that it ends up between $P_j(t+1)$ and a cell that is an element of $N_j(t+1)$.

Pair 4 $\{N_i(t), P_j(t)\}$: if $N_i(t)$ contains a cell that shares the same pin as $P_j(t)$, then at time t, D_i will drift between $P_i(t)$ and the cell of interest in $N_i(t)$. If this is not the case, D_i may not be able to move to $P_j(t+1)$ because it will no longer overlap with $P_j(t+1)$.

Pair 5 $\{N_i(t), P_j(t+1)\}$: if $N_i(t)$ contains a cell that shares the same pin as $P_j(t+1)$, D_i will not move properly at time $t+1$ unless $P_i(t+1)$ is the cell in $N_i(t)$ that shares the same pin as $P_j(t+1)$.

Pair 6 $\{N_i(t+1), P_j(t+1)\}$: if $N_i(t+1)$ contains a cell that shares the same pin as $P_j(t+1)$ then

at time t+1, D_i will drift between $P_i(t+1)$ and the cell of interest in $N_i(t+1)$.

We have analytically confirmed that the control pins of these six pairs must be mutually exclusive to prevent interference between droplets D_i and D_j. These lead to the following necessary and sufficient conditions, called the interference constraints:

1. $\bar{k}\left(\{P_i(t)\}\right)\cap\bar{k}\left(N_j(t)\right)=\varnothing$
2. $\bar{k}\left(\{P_i(t+1)\}\right)\cap\bar{k}\left(N_j(t+1)\right)=\varnothing$
3. $\bar{k}\left(N_i(t)\right)\cap\bar{k}\left(\{P_j(t)\}\right)=\varnothing$
4. $\bar{k}\left(N_i(t+1)\right)\cap\bar{k}\left(\{P_j(t+1)\}\right)=\varnothing$
5. $\bar{k}\left(\{P_i(t+1)\}\right)\cap\bar{k}\left(N_j(t)-\{P_j(t+1)\}\right)=\varnothing$
6. $\bar{k}\left(\{P_j(t+1)\}\right)\cap\bar{k}\left(N_i(t)-\{P_i(t+1)\}\right)=\varnothing$

Moving one droplet and stalling the other is a special case that can be used when concurrent movement of two droplets leads to interference, i.e., violation of the interference constraints. In this situation, P_j=$P_j(t)$=P_j (t+1) and the interference constraints reduce to

1. $\bar{k}\left(\{P_i(t)\}\right)\cap\bar{k}\left(N_j\right)=\varnothing$;
2. $\bar{k}\left(\{P_i(t+1)\}\right)\cap\bar{k}\left(N_j\right)=\varnothing$;
3. $\bar{k}\left(N_i(t)\right)\cap\bar{k}\left(\{P_j\}\right)=\varnothing$;
4. $\bar{k}\left(N_i(t+1)\right)\cap\bar{k}\left(\{P_j\}\right)=\varnothing$.

Let $P_i(\bullet)$ denote the cell position of D_i. The fluidic constraints that were discussed in Section 3.2 can be expressed as:

1. $\left|P_{i,x}(t+1)-P_{j,x}(t+1)\right|\geq 2$ or $\left|P_{i,y}(t+1)-P_{j,y}(t+1)\right|\geq 2$;
2. $\left|P_{i,x}(t+1)-P_{j,x}(t)\right|\geq 2$ or $\left|P_{i,y}(t+1)-P_{j,y}(t)\right|\geq 2$;
3. $\left|P_{i,x}(t)-P_{j,x}(t+1)\right|\geq 2$ or $\left|P_{i,y}(t)-P_{j,y}(t+1)\right|\geq 2$.

The subscript x denotes the x-component of the position (column #) and the subscript y denotes the y-component of the position (row #). The first two constraints are identical if we set $P_j(t)$ = $P_j(t+1)$, which is true for the case of one droplet moving and one droplet waiting. Constraint 3 is not necessary if the diagonal adjacent effect is negligible. The following are alternative statements for the fluidic constraints:

$$(1)\left\{P_i(t+1)\right\}\cap N_j\left(t+1\right)=\varnothing;\quad (2)\left\{P_i(t+1)\right\}\cap N_j\left(t\right)=\varnothing;\quad (3)\left\{P_j\left(t+1\right)\right\}\cap N_i\left(t\right)=\varnothing.$$

The interference constraints are designed to prevent "long-range" interference between the intended paths of droplets whereas the fluidic constraints are necessary to avoid "short-range" interference in the form of inadvertent mixing. This is because interference is a manifestation of control pin sharing between cells anywhere on the array while mixing (i.e., when fluidic constraints are violated) is a result of physical contact between droplets.

Given k independent pins, our goal is to maximize the number of independent movements that a droplet can undertake from each position of the array while not interfering with another droplet on the same array. If interference results, some droplets can be forced to wait (stall) while others move. If interference still occurs, then there is no safe way to transport the droplets on the same array along their desired paths unless the schedule/routes are modified. In order to develop a useful, application-independent index that represents the independence of movement for two droplets on an array that can be easily extended to multiple droplets, we need a metric for comparison. If every cell has a dedicated control pin, there is no possibility of interference

between droplets. Hence, only the fluidic constraints need to be considered.

For simplicity, we first consider the case of one droplet moving while the other droplet waits. Let Φ be the set of all possible pin configurations using k pins for an $n \times m$ array. For a particular pin configuration $c \in \Phi$ using k-pins in our 2-droplet system, let $I_c(k,n,m)$ be the pin-constrained non-interference index (PCNI). It takes on a value between 0 and 1 and equals the fraction of legal moves for two droplets (one moving, one waiting) on an $n \times m$ array with each cell having its own dedicated control pin that are still legal with pin layout $c \in \Phi$ and $k < n \times m$ pins, i.e.,

$$I_c(k,n,m) = \frac{number\ of\ legal\ moves\ in\ configuration\ c\ with\ k < n \times m\ pins}{number\ of\ legal\ moves\ with\ k = n \times m\ pins}$$

The interference and fluidic constraints discussed above can be used to determine $I_c(k,n,m)$ for a given pin layout. The computational complexity of this procedure is $O(n^2m^2)$.[6] For example, consider the 3 x 4 array with the 9-pin layout shown in Figure 13(a). Using the interference and fluidic constraints, we find that there are a total of 48 legal moves for the given array dimensions and pin layout. A listing of all legal moves is provided in Table II. The reference pin layout, which provides full control with 12 independent pins, is shown in Figure 13(b). To determine the number of legal moves for this reference pin layout, only the fluidic constraints need to be considered. The reference pin layout yields 148 legal moves. Therefore, the fraction of legal moves for the reference pin layout that are still legal for the proposed pin layout is 48/148 = 0.324. Two other layouts and their PCNIs are illustrated in Figures 13(c) and 13(d). Better layouts seem to loosely obey two principles: 1) spread out the placement of pins that are used multiple times, and 2) place multiply-used pins on cells that have fewer neighbors (e.g., sides and corners).

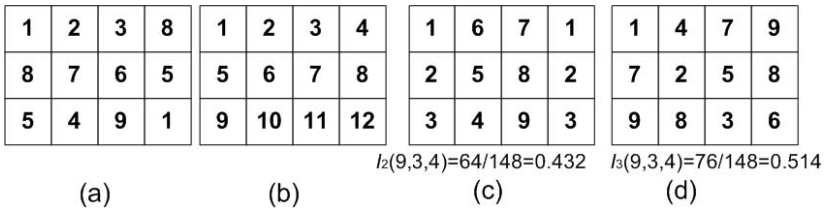


Figure 13. (a) 9-pin layout, (b) reference pin layout for a 3x4 array, (c) alternative 9-pin layout, (d) second alternative 9-pin layout.

For the case of two droplets moving concurrently, we use the same definition as before for the PCNI, except that a legal move is now characterized by a unique quartet, ($P_i(t)$, $P_i(t+1)$, $P_j(t)$, $P_j(t+1)$), that satisfies the corresponding fluidic and interference constraints. For example, for the same 3x4 array with the pin layout shown in Figure 13(a), there are a total of 88 legal moves. The reference pin layout, shown in Figure 13(b), yields 264 legal moves. Therefore, the fraction of legal moves for the reference pin layout that are still legal for the proposed pin layout is simply 88/264 = 0.333.

[6] See Appendix A for the algorithm.

Table II. Listing of all 48 legal moves for the pin layout shown in Figure 13(a) assuming that D_i moves while D_j waits.

$P_i(t)$	$P_i(t+1)$	P_j		
(1,1)	(1,2)	(3,1)	(3,2)	
	(2,1)	(2,3)		
(1,2)	(1,1)	(3,1)	(3,2)	
	(2,2)	(2,4)		
	(1,3)	(3,1)	(3,2)	(3,3)
(1,3)	(1,2)	(3,1)	(3,2)	(3,3)
	(2,3)	(1,1)		
	(1,4)	(3,2)	(3,3)	(3,4)
(1,4)	(1,3)	(3,2)	(3,3)	(3,4)
	(2,4)	(1,2)		
(2,1)	(1,1)	(2,3)		
	(2,2)	NONE		
	(3,1)	(3,3)		
(2,2)	(1,2)	(2,4)		
	(2,3)	NONE (all placements violate fluidic constraints)		
	(2,1)	NONE		
	(3,2)	(3,4)		
(2,3)	(1,3)	(1,1)		
	(2,2)	NONE (all placements violate fluidic constraints)		
	(2,4)	NONE		
	(3,3)	(2,1)		
(2,4)	(1,4)	(1,2)		
	(2,3)	NONE		
	(3,4)	(2,2)		
(3,1)	(2,1)	(3,3)		
	(3,2)	(1,1)	(1,2)	(1,3)
(3,2)	(3,1)	(1,1)	(1,2)	(1,3)
	(2,2)	(3,4)		
	(3,3)	(1,2)	(1,3)	(1,4)
(3,3)	(3,2)	(1,2)	(1,3)	(1,4)
	(2,3)	(2,1)		
	(3,4)	(1,3)	(1,4)	
(3,4)	(2,4)	(2,2)		
	(3,3)	(1,3)	(1,4)	

5.2. Virtual Partitioning Scheme

An alternative method of controlling multiple droplets on a single array is to "virtually" partition the array into regions. At any given time, partitions use non-overlapping sets of pins. Recall that regardless of size, a two-dimensional array needs a minimum of $k = 5$ pins to have full control of a single droplet when the diagonal adjacent effect is neglected. If there is a maximum of one droplet in each partition at all times, then interference is not possible.

With the partitioned array, the number of droplets that can be transported simultaneously *without* stall cycles is equal to the number of partitions since partitions do not share any control pins. Fluidic constraints must also be satisfied so that inadvertent mixing does not occur. The only time when droplets in each partition need to be near each other is when they are to be mixed as part of the bioassay protocol. We therefore propose to create central partition(s) of the array

for mixing purposes, as shown in Figure 14. When mixing is complete, the merged droplet can be moved to the appropriate partition for further processing without interference with other droplets.

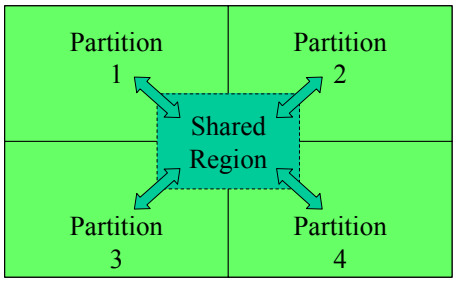


Figure 14. Schematic of partitioned array with a shared region.

As the number of droplets increases, the control bandwidth for an array based on the partitioned layout increases gradually with respect to the fixed $k = 5$, which is the minimum number of pins needed for a non-partitioned pin layout to yield full control of a single droplet regardless of array size. However, the *absence of interference* becomes more and more significant relative to the non-partitioned pin layout and usually outweighs the reduced flexibility incurred from the partitions. Moreover, another distinct advantage of the partitioned pin layout is that all droplets can move simultaneously since there is almost never a need for a droplet to wait. However, one disadvantage of a partitioned system of pins is the reduction in reconfigurability. This is especially true when there are designated central regions that are used for mixing. If one of these mixing regions has a defect, relocation will not be as straightforward as with the non-partitioned pin layout array. We can enable dynamic rearrangement of the partitions as described in the next section, but fixed partitions are satisfactory for disposable chips that run the same set of assays throughout its lifetime.

5.2.1. Impact of Partitioning on PCNI

The 5x5 array shown in Figure 15(a) has a 10-pin layout. A cursory examination shows that even a single droplet will suffer from interference on this array. For example, any droplet in columns 2, 3, or 4 will be confined to its current column—horizontal movement is impossible. The problem is that the interior cells of the array are not surrounded by unique neighbors. To address this issue, we vertically flip the third and fourth columns and then swap the control pins of cells (4,2) and (4,3) with (2,2) and (2,3) respectively, as shown in Figure 15(b).

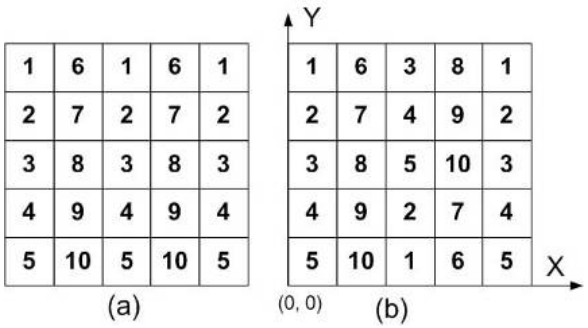


Figure 15. (a) A 10-pin layout and (b) another 10-pin layout for a 5x5 array.

With this modified layout, every cell has a set of uniquely controlled nearest neighbors,

allowing for unhindered movement of a single droplet. This new arrangement is characterized by a two-droplet one-move/one-wait PCNI of 0.2626. For most assays, we can dynamically divide the array into two partitions to accommodate the two droplets such that they will never interfere.

Once the partitions are determined and the electrical connections between the electrodes and the external pins are laid out on the substrate (or the PCB), we have a hardwired platform. While the partitions can be determined appropriately for one or more bioassays, the effectiveness of these partitions may be reduced if a user has to map other assay protocols onto the same partition configuration. A switch network can be used to dynamically reconfigure the interconnections between the k electrical signals and the n x m signals that drive the electrode array as shown in Figure 16(a). The control signals can either be derived from a computer-controlled set up, or the switch network and the control signals can be implemented using a field-programmable gate-array. These partitions can either be changed between sets of concurrent assays, or at appropriate "breakpoints" during assay execution. An example 2-partition layout utilizing 10 pins is shown in Figure 16(b). This partitioned array has a PCNI of 0.4041, a 53.8842% increase relative to the non-partitioned array.

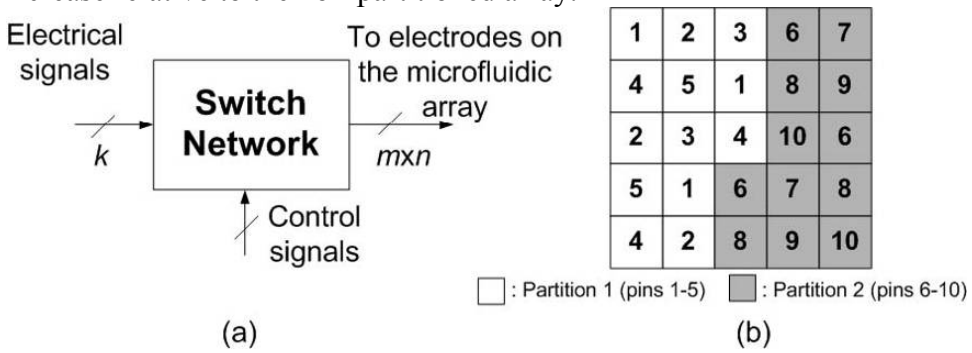


Figure 16. (a) Dynamically-reconfigurable interconnection network; (b) a 2-partition layout for a 5x5 array.

In general, if we partition an array such that at any time during an assay, at most one droplet occupies each partition, we can guarantee that the assay will be completed at the same throughput as the fully-addressable array with $k = n \times m$. However, while this condition is sufficient for achieving the same throughput as a non-partitioned array, it is not necessary. Computation of the PCNI is no different for partitioned arrays relative to non-partitioned arrays; hence, the increases in PCNI observed for partitioned arrays do not involve adherence to the rule of at most one droplet per partition at all times.

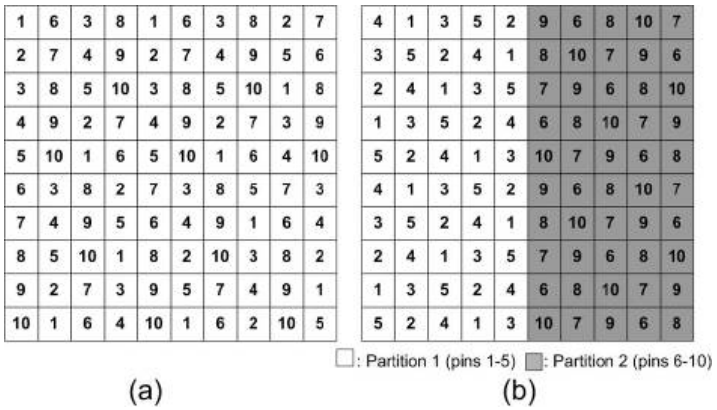


Figure 17. (a) A 10-pin non-partitioned layout and (b) a partitioned layout for a 10x10 array.

Consider the non-partitioned 10x10 array with $k = 10$ shown in Figure 17(a). The non-interference index is 0.1467. If we instead divide the array into two partitions, still with $k = 10$, then one possible layout is shown in Figure 17(b). The non-interference index increases to 0.4565 for the 2-partition array, an augmentation of 211.18% relative to the non-partitioned array. In general, as the array size increases, and the number of pins and number of partitions remains constant, the PCNI for a partitioned array will increase while that of a non-partitioned array will decrease. The reasons for this are the following: (1) For a non-partitioned array, the possibilities for interference increase as array size increases and pins number remains the same, and (2) For a partitioned array, the partition size increases with the array size if k and the number of partitions are constant. Larger partitions mean that the proportion of movement pairs that violate fluidic constraints at the borders decreases, while interference is rarely a concern for partitioned arrays.

Figure 18(a) shows the same 10x10 array with the two partitions rearranged. The index is 0.4215, which is slightly less than the index value of 0.4565 obtained with the vertical-boundary partition scheme shown in Figure 17(b).

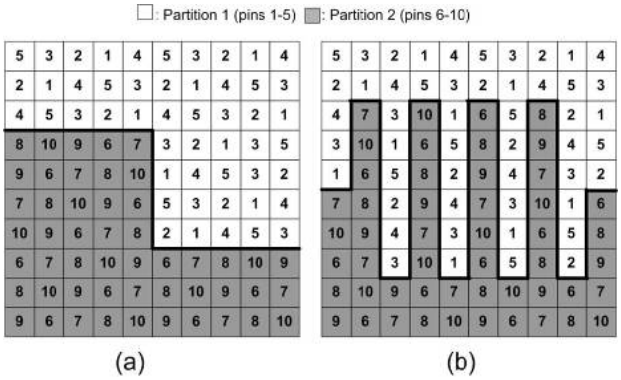


Figure 18. (a) Another partitioned pin layout for a 10x10 array; (b) a partitioned pin layout for a 10x10 array with a long boundary length.

It is interesting to examine why the second partitioning scheme result in a lower PCNI. The boundary length between the partitions dictates the number of opportunities for fluidic interference (i.e., inadvertent mixing) between two droplets even when they satisfy the "rule" of at most one droplet per partition at any given time. The boundary length of the first partitioning scheme is 10 edges whereas the boundary length of the second is 14 edges. To illustrate this further, the example in Figure 18(b) also has two partitions for a 10x10 array with $k = 10$. The boundary length is 58. As expected, the PCNI drops significantly to 0.2478 with the large increase in boundary length, but it still represents a 68.92% increase relative to the non-partitioned array. These observations lead us to conclude that, to maximize the PCNI, the boundary length between partitions should be kept to the minimum possible for a given assay.

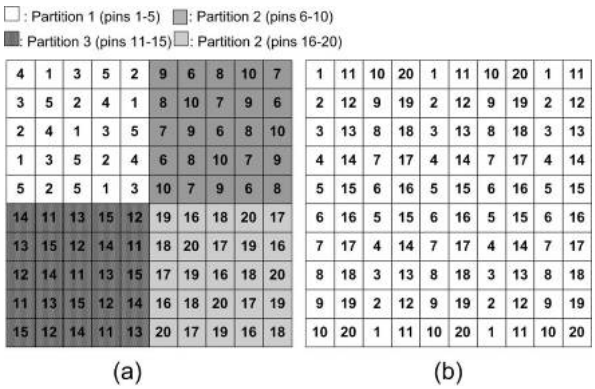


Figure 19. (a) A partitioned pin layout for a 10x10 array with k = 20 and 4 partitions. (b) A non-partitioned pin layout for a 10x10 array.

To illustrate the use of multiple partitions, consider the array pin layout with k = 20 and 4 partitions, as shown in Figure 19(a). The two droplet PCNI is 0.7331 relative to the non-partitioned array shown in Figure 19(b), which yields an index of 0.6772. It is important to note that since the PCNI reported here is only computed for two droplets simultaneously on the array, the advantages of the 4-partition layout in Figure 19(a) are not being fully exploited. Whereas four droplets maintained in unique partitions would have no potential for interference and therefore the four-droplet PCNI will only drop slightly for the partitioned array relative to the two-droplet PCNI, the four-droplet PCNI will drop precipitously for the non-partitioned array. It is remarkable that even if we ignore this obvious advantage, the two-droplet PCNI of the partitioned array is still 8.25% higher than that of the non-partitioned array. Even though the virtual partitions are not considered in the computation of the PCNI, the virtual partitioning scheme still leads to significantly higher indices relative to non-partitioned arrays. Therefore, it is generally advantageous to partition the array even if the assays of interest use the array in a random fashion, i.e., with no regard to the partitions.

5.2.2. Evaluation Example: Multiplexed Bioassays

To show how the proposed virtual partitioning method can be used for pin-constrained microfluidic biochips, we use a real-life experiment of a multiplexed biochemical assay consisting of a glucose assay and a lactate assay based on colorimetric enzymatic reactions, which have been demonstrated recently [6].

The digital microfluidic biochip contains a 15×15 microfluidic array, as shown in Figure 20. The schedule for the set of bioassays, if a fully-addressable array with 225 control pins is available, is listed in Table III; one iteration of the multiplexed assays takes 25.8 seconds. The movement of droplets (including test droplets) is controlled using a 50 V actuation voltage with a switching frequency of 16 Hz. A depiction of the droplet paths for multiplexed glucose and lactase assays is illustrated in Figure 20.

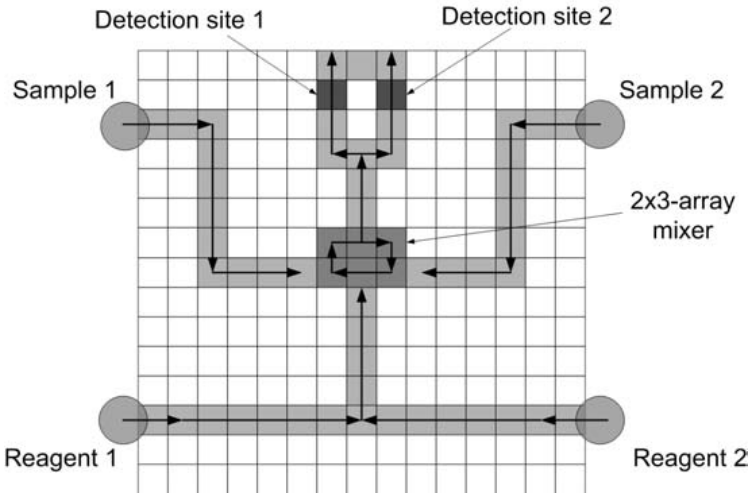


Figure 20. A 15×15 array used for multiplexed bioassays.

Table III. Bioassay schedule for fully-addressable array.

Step/Time Elapsed (s)	**Operation**
Step 1 / 0	Sample (S) 2 and reagent (R) 2 start to move towards the mixer.
Step 2 / 0.8	S2 and R2 begin to mix together and turn around in the 2×3-array mixer.
Step 3 / 6.0	S1 and R1 start to move towards the mixer. S2 and R2 continue to mix.
Step 4 / 6.8	S2 and R2 finish the mixing and product (P) 2 leaves the mixer to optical detection location 2. S1 and R1 begin to mix in the mixer.
Step 5 / 12.8	S1 and R1 finish the mixing and P1 leaves the mixer to the optical detection location 1. P2 continues the absorbance detection.
Step 6 / 19.8	P2 finishes optical detection & leaves the array to waste reservoir. P1 continues the absorbance detection.
Step 7 / 25.8	P1 finishes optical detection and leaves the array to the waste reservoir. One procedure of the multiplexed bioassays ends.

As with most assays, partitioning of the pin layout is effective in reducing the input bandwidth while maintaining the same throughput. Five partitions are satisfactory in preventing interference between multiple droplets on the array, as shown in Figure 21. Since only five control pins are necessary for full control of a single droplet within each partition, only 25 out of the possible 225 control pins are necessary, i.e., 11.11%. This represents a significant reduction in input bandwidth without sacrificing throughput. Note that the mixing region (i.e., pins 6-10) is designated such that as soon as Sample and Reagent enter the mixer, they merge to form one droplet so that there is no need to worry about multiple droplets in the mixer and the potential interference, precluding the need for a dedicated mixing region.

What if the 15 x 15 array used 25 control pins without a partitioning scheme? There would be many more chances of interference and the throughput would be greatly reduced. To illustrate this, we examine a similar assay on a smaller array that can be easily analyzed by hand. Figure 22 displays a 5 x 5 array that has a fully addressable reference layout. The assay schedule is shown in the chart.

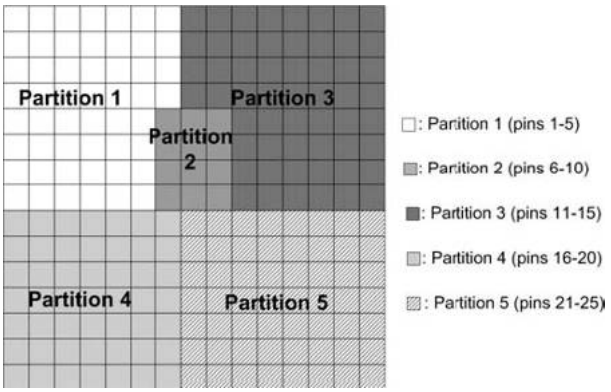


Figure 21. A partitioned 15×15 microfluidic array.

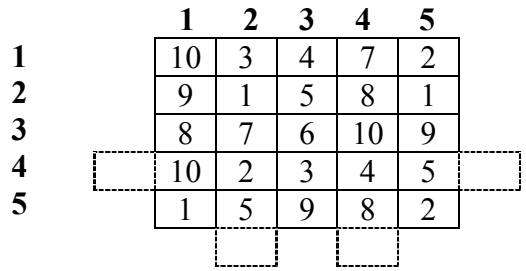
Reference Layout

	1	2	3	4	5
1	1	10	11	20	21
2	2	9	12	19	22
3	3	8	13	18	23
4	4	7	14	17	24
5	5	6	15	16	25

Time	Action	P_i	P_j	P_k
1	Dispense	(4,1)	(4,5)	-
2	Move	(4,2)	(4,4)	-
3	Merge	(4,3)	(4,3)	-
4	Mix	(4,4)	(4,4)	-
5	Mix	(3,4)	(3,4)	-
6	Mix	(3,3)	(3,3)	-
7	Mix, Dispense Dk	(3,2)	(3,2)	(4,5)
8	Split, Move	(2,2)	(4,2)	(4,4)
9	Detect, Merge	(1,2)	(4,3)	(4,3)
10	Mix		(4,2)	(4,2)
11	Mix		(3,2)	(3,2)
12	Mix		(3,3)	(3,3)
13	Mix		(3,4)	(3,4)
14	Split		(2,4)	(4,4)
15	Detect		(1,4)	(5,4)
16	Dispose			into waste

Figure 22. Fully-addressable 5 x 5 layout and assay schedule.

Now suppose we are constrained to only having 10 pins. Figure 23 shows one possible layout that has a PCNI of 0.169811.



Time	Action	Pins Active	P_i	P_j	P_k
1	Dispense	5,10	(4,1)	(4,5)	-
2	Stall Di, Move Dj	4,10	(4,1)	(4,4)/(3,4) – minor interference	-
3	Move Di, Stall Dj	2,4	(4,2)	(4,4)	-
4	Merge	4	(4,3)	(4,3)	-
5	Mix	4	(4,4)	(4,4)	-
6	Mix	10	(3,4)	(3,4)	-
7	Mix	6	(3,3)	(3,3)	-
8	Mix	7	(3,2)	(3,2)	-
9	Dispense Dk	5,7	(3,2)	(3,2)	(4,5)
10	Move Dk	4,7	(3,2)	(3,2)	(4,4)
11	Split Dij	1,2,4	(2,2)	(4,2)	(4,4)
12	Detect, Merge	3	(1,2)	(4,3)	(4,3)
We cannot move D_{jk} since D_i needs to stay on the detector. Suppose detection takes 10 time steps.					
23	Mix, Dispose of Dij	2	-	(4,2)	(4,2)
24	Mix	7	-	(3,2)	(3,2)
25	Mix	6	-	(3,3)	(3,3)
26	Mix	10	-	(3,4)	(3,4)
27	Split	4.8	-	(2,4)	(4,4)/(5.4)
28	Move	4,7	-	(1,3)/(1,4)	(4,4)
29	Move	7,8	-	(1,4)/(2,4)	(5,4)

At least two extra time steps are added to the second detection because the drift interference delays initiation of the detection step.

Figure 23. 10-pin constrained 5 x 5 layout and effect on assay schedule.

In total, the assay takes almost twice as long with the pin-constrained layout without partitioning. In a fashion similar to the 15 x 15 array illustrated in Figure 21, partitioning can be used to negate the possibility of interference on the 5 x 5 array for this particular assay (Figure 24). Notice that there is really no need for a dedicated mixing region for this particular assay. When droplets from the pink partition cross into the green partition, merging occurs immediately so there is never more than one droplet in a partition. The throughput of this 9-pin layout is the same as that of the reference layout (Figure 22), which uses 25 control pins.

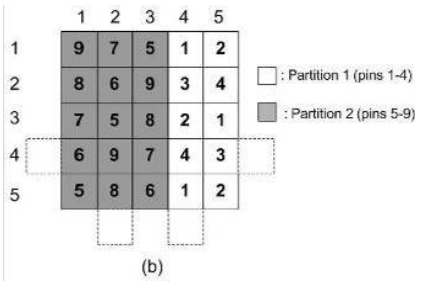


Figure 24. Partitioned 5 x 5 layout using 9 pins.

6. Conclusions

In this paper, we addressed three important problems in automating the design and testing of digital microfluidic biochips. First, we developed the first systematic routing method for digital microfluidic biochips; our approach attempts to minimize the number of cell used for droplet routing, while satisfying constraints imposed by performance goals and fluidic properties. We first formulated the droplet routing problem, where the total number of cells used for routing serves as the objective criterion. Important constraints imposed by performance goals and fluidic properties have also been incorporated. A detailed experimental validation has been carried out for the fluidic constraint rules. Based on this problem formulation, a two-stage routing method has been proposed; this method is independent of the routing order of nets. We have also exploited the features of dynamic reconfigurability and independent controllability of electrodes to modify droplet pathways to override potential violation of fluidic constraints. This method will next be integrated with architectural-level synthesis and module placement to form a comprehensive synthesis tool for digital microfluidic biochips.

Next, we described recent experiments that reveal the inadequacy of testing based on Hamiltonian paths in a graph model of the microfluidic array and introduced a novel testing methodology for localizing faults. The previous approaches of mapping the droplet flow path problem to that of finding a Hamiltonian path in a graph model of the array is not sufficient to detect electrode-short and fluidic-open faults that affect two adjacent electrodes. To detect these edge-dependent defects, we developed a testing method based on an adaptation of Fleury's algorithm to find Euler circuits in the non-directed graph representation of microfluidic arrays.

Finally, we proposed a new design automation technique for pin-constrained digital microfluidic biochips that dramatically reduces the input bandwidth between the electronic controller and the microfluidic array while minimizing any decrease in performance. We first formulated the problem of determining the minimum number of control pins for full control of a single droplet. The interference problem for multiple droplets on a pin-constrained biochip has also been investigated. A new method called virtual partitioning has been proposed to reduce or remove interference in pin-constrained biochips. The real-life example of a set of multiplexed bioassays has been used to evaluate the effectiveness of the proposed method. As part of ongoing work, we are developing an algorithm to automatically find optimal pin layouts with the maximum PCNI for a given number of control pins. We are also creating a trace-based partitioning algorithm to determine suitable partitioning schemes given k pins, an $n \times m$ array, and assay schedules. By dramatically reducing the number of control pins with minimal impact on assay throughput, the proposed design method is expected to reduce cost and lead to the miniaturization of mixed-technology disposable biomedical devices for the emerging healthcare market.

7. Acknowledgements

I would like give my sincerest gratitude to Professor Krishnendu Chakrabarty and Professor Richard Fair for welcoming me into their research groups, for their invaluable guidance, inspiration, encouragement, and ever willingness to help me. I greatly appreciate the time Dr. Fei Su spent working with me despite being busy finishing his own program. I would also like to thank Vladi Ivanov, Phil Paik, and Vijay Srinivasan for helping me with the laboratory setups and methods. Together they made this one of the most enjoyable and rewarding research experiences I have ever had.

8. References

[1] F. Su, W.L. Hwang, and K. Chakrabarty, "Droplet routing in the synthesis of digital microfluidic biochips," *Proc. Design, Automation, and Test in Europe Conference,* 2006.

[2] F. Su, W.L. Hwang, A. Mukherjee, and K. Chakrabarty, "Defect-oriented testing and diagnosis of digital microfluidics-based biochips," *Proc. IEEE International Test Conference,* 2005.

[3] W.L. Hwang, F. Su, and K. Chakrabarty, "Automated design of pin-constrained digital microfluidic arrays for lab-on-a-chip applications," accepted for publication in *Proc. IEEE/ACM Design Automation Conference,* 2006.

[4] E. Verpoorte and N. F. De Rooij, "Microfluidics meets MEMS", *Proceedings of the IEEE*, vol. 91, pp. 930-953, 2003.

[5] T.H. Schulte et al., "Microfluidic technologies in clinical diagnostics", *Clinica Chimica Acta*, vol. 321, pp. 1-10, 2002.

[6] V. Srinivasan et al., "An integrated digital microfluidic lab-on-a-chip for clinical diagnostics on human physiological fluids," *Lab on a Chip*, pp. 310-315, 2004.

[7] Infineon Electronic DNA Chip, http://www.infineon.com

[8] Nanogen NanoChip®, http://www.nanogen.com

[9] M.G. Pollack et al., "Electrowetting-based actuation of liquid droplets for microfluidic applications", *Applied Physics Letters*, vol. 77, pp. 1725-1726, 2000.

[10] S.K. Cho et al., "Toward digital microfluidic circuits: creating, transporting, cutting and merging liquid droplets by electrowetting-based actuation", *Proc. IEEE MEMS Conf.*, pp. 32-52. 2002.

[11] P.Y. Paik et al., "Rapid droplet mixers for digital microfluidic systems", *Lab on a Chip*, vol. 3, pp. 253-259, 2003.

[12] F. Su and K. Chakrabarty, "Architectural-level synthesis of digital microfluidics-based biochips", *Proc. IEEE Intl. Conf. on CAD*, pp. 223-228, 2004.

[13] F. Su and K. Chakrabarty, "Design of fault-tolerant and dynamically-reconfigurable microfluidic biochips", *Proc. DATE Conference*, pp.1202-1207, 2005.

[14] J. Gong, and C.J. Kim, "Two-dimensional digital microfluidic system by multi-layer printed circuit board," *Proc. of IEEE MEMS 2005*, pp. 726-729, 2005.

[15] P.Y. Paik et al., "Coplanar digital microfluidics using standard printed circuit board processes", *Proc. Intl. Conf. on MicroTAS*, pp. 566-568, 2005.

[16] T. Mukherjee and G.K. Fedder, "Design methodology for mixed-domain systems-on-a-chip

[DMEMS design]", *Proc. IEEE VLSI System Level Design*, pp. 96 - 101, 1998.

[17] J. Zeng and T. Korsmeyer, "Principles of droplet electrohydrodynamics for lab-on-a-chip", *Lab on a Chip*, vol. 4, pp. 265-277, 2004.

[18] T. Korsmeyer et al., "Design tools for bioMEMS", *Proc. DAC*, pp. 622-627, 2004.

[19] R.T. Hadsell and P.H. Madden, "Improved global routing through congestion estimation", *Proc. DAC*, pp. 28-21, 2003.

[20] N. Sherwani, *Algorithms for VLSI Physical Design Automation*, Kluwer Academic Publishers, MA 1995.

[21] S. Sait and H. Youssef, *VLSI Physical Design Automation: Theory and Practice*, IEEE Press, NY, 1995.

[22] C. Sechen, *VLSI Placement and Global Routing Using Simulated Annealing*, Kluwer Academic Publishers, Boston, MA, 1988.

[23] A.J. Pfeiffer et al., "Simultaneous design and placement of multiplexed chemical processing systems on microchip", *Proc. ICCAD*, pp. 229-236, 2004.

[24] E. Griffith and S. Akella, "Coordinating multiple droplets in planar array digital microfluidics systems", *Workshop on the Algorithmic Foundations of Robotics*, 2004.

[25] P. Paik et al., "Rapid droplet mixers for digital microfluidic systems", *Lab on a Chip*, vol. 3, pp. 253-259, 2003.

[26] F. Su et al., "Testing of droplet-based microelectrofluidic systems", *Proc. IEEE Int. Test Conf.*, pp. 1192-1200, 2003.

[27] F. Su et al, "Test planning and test resource optimization for droplet-based microfluidic systems", *Proc. IEEE Eur. Test Sym.*, pp. 72-77, 2004.

[28] F. Su et al., "Concurrent testing of droplet-based microfluidic systems for multiplexed biomedical assays", *Proc. IEEE Int. Test Conf.*, pp. 883-892, 2004.

[29] Douglas B. West, Introduction to Graph Theory, Prentice Hall, NJ, 1996.

[30] S.K. Fan et al., "Manipulation of multiple droplets on N×M grid by cross-reference EWOD driving scheme and pressure-contact packaging", *Proc. IEEE MEMS Conf.*, pp. 694-697, 2003.

[31] J. Gross and J. Yellen, *Graph Theory and its Applications*, Boca Raton, FL: CRC Press, 1999.

[32] C.H. Papadimitriou, *Computational Complexity*, Reading, MA: Addison Wesley, 1993.

Appendix A
PCNI Algorithms

Conjecture 1: There exists a configuration, c*$\in\Phi$, such that $I_{c^*}(k,n,m) \geq I_c(k,n,m)$ for all c$\in\Phi$ in our 2-droplet system. Because of symmetry of the $n\times m$ array, c* may not be unique. We call c* an "optimal" pin-configuration of k pins for an $n\times m$ array and $I^*(k,n,m) = I_{c^*}(k,n,m)$, the maximum achievable pin-constrained non-interference index.

Conjecture 2: For a fixed $n\times m$ *array* with an "optimal" configuration c* using k pins, $I^*(k,n,m)$ is monotonically increasing with k where $k \leq n\times m$, making it a good metric for measuring potential interference. That is, $I_{c^*}(k,n,m) \leq I_{c^*}(s,n,m)$ for k≤s. In other words, a greater number of control pins for the same size array cannot lead to a lower non-interference index.

The algorithm is implemented in C++ and operates in two modes: (1) Determine the pin-constrained non-interference index for a pin-layout specified in a file (2) Find an optimal layout and its degeneracy given k, n, and m. The lines in bold are pseudocode and may involve multiple lines of actual code. A legal move is characterized by a unique triplet ($P_i(t)$, $P_i(t+1)$, P_j) that satisfy the fluidic and interference constraints.

```
int main()
{
   int count = 0, countref = 0;     // count holds the total number of legal
                                    moves without interference for two
                                    droplets on user defined pin layout;
                                    countref holds total number of legal
                                    moves for pin layout in which all cells
                                    have a dedicated pin
   Obtain array dimensions (n x m) and pin layout from user, store pin layout
      in array, pinLayout
   Using array dimensions, create array for reference pin layout (all cells
      have dedicated pins), pinLayoutRef
   count = countLegalMoves(pinLayout);
   countref = countLegalMoves(pinLayoutRef);

   float index = count/countref;
}
int countLegalMoves(array pinLayout)
{
   int count;
   for (int i = 0; i < n; i++)      // Iterate through all rows
   {
      for (int j = 0; j < m; j++)   // Iterate through all columns
      {
         Set Pi(t) to coordinate (i,j)
         Determine all possible Pi(t+1) and store into vector, next
            int len = length(next);
         for (int k = 0; k < len; k++)    // Iterate through all Pi(t+1)
         {
            Set Pi(t+1) to next(k)
            for (int a = 0; a < n; a++)   // Iterate through rows
            {
               for (int b = 0; b < m; b++) //Iterate through columns
```

```
                {
                    Set P_j to coordinate (a,b)
                    Check all interference and fluidic constraints
                    If placement of D_j is legal, then increment count
                }
            }
        }
    }
}
return count;
}
```

The following algorithm can be applied to obtain a pin-constraint independence index (array has n rows and m columns). The output value, *index*, is a value between 0 and 1 that is the fraction of legal moves for two droplets (both moving) on an *n* x *m* array with each cell having its own dedicated control pin that are still legal with a different pin layout using $k < n\ x\ m$ pins. The lines in bold are pseudocode and may involve multiple lines of actual code.

```
int main()
{

   int count = 0, countref = 0;      // count holds the total number of legal
                                     moves without interference for two
                                     droplets on user defined pin layout;
                                     countref holds total number of legal
                                     moves for pin layout in which all cells
                                     have a dedicated pin

   Obtain array dimensions (n x m) and pin layout from user, store pin layout
      in array, pinLayout

   Using array dimensions, create array for reference pin layout (all cells
      have dedicated pins), pinLayoutRef

   count = countLegalMovesBoth(pinLayout);
   countref = countLegalMovesBoth(pinLayoutRef);

   float index = count/countref;
}

int countLegalMovesBoth(array pinLayout)
{
   int count;
   for (int i = 0; i < n; i++)       // Iterate through all rows
   {
      for (int j = 0; j < m; j++)    // Iterate through all columns
      {
         Set P_i(t) to coordinate (i,j)
         Determine all possible P_i(t+1) and store into vector, next_i

         int len_i = length(next);

         for (int k = 0; k < len_i; k++)  // Iterate through all P_i(t+1)
         {
```

```
            Set P_i(t+1) to next_i(k)

            for (int a = 0; a < n; a++)   // Iterate through rows
            {
               for (int b = 0; b < m; b++) //Iterate through columns
               {
                   Set P_j to coordinate (a,b)
                   Determine all possible P_j(t+1); store into vector, next_j
                   int len_j = length(next_j);

                   for (int r = 0; r < len_j; r++)
                   {
                        Set P_j(t+1) to next_j(r)
                        Check all interference and fluidic constraints
                        If placement of D_j is legal, then increment count
                   }
               }
            }
         }
      }
   }
   return count;
}
```