

# Integrated Fluidic-Chip Co-Design Methodology for Digital Microfluidic Biochips

Jia-Wen Chang, Sheng-Han Yeh, Tsung-Wei Huang, and Tsung-Yi Ho

**Abstract**—Recently, digital microfluidic biochips (DMFBs) have revolutionized many biochemical laboratory procedures and received much attention due to their many advantages, such as high throughput, automatic control, and low cost. To meet the challenges of increasing design complexity, computer-aided-design (CAD) tools have been used to build DMFBs efficiently. Current CAD tools generally conduct a two-stage based design flow of fluidic-level synthesis followed by chip-level design to optimize fluidic behaviors and chip architecture separately. Nevertheless, existing fluidic-chip design gap will become even wider with a rapid escalation in the number of assay operations incorporated into a single DMFB. As more and more large-scale assay protocols are delivered in the current emerging marketplace, this problem may potentially restrict the effectiveness and feasibility of the entire DMFB realization and thus needs to be solved quickly. In this paper, we propose the first fluidic-chip co-design methodology for DMFBs to effectively bridge the fluidic-chip design gap. Our work provides a comprehensive integration throughout fluidic-operation scheduling, chip layout generation, control pin assignment, and wiring solution to achieve higher design performance and feasibility. Experimental results show the effectiveness, robustness, and scalability of our co-design methodology on a set of real-life assay applications.

**Index Terms**—Biochip, co-design, digital microfluidics, integer linear programming (ILP).

## I. INTRODUCTION

DIGITAL microfluidic biochips (DMFBs), a more versatile category of microfluidic technology, have recently emerged as a popular alternative for laboratory experiments. Compared to conventional bench-top procedures, DMFB technology offers advantages of low sample and reagent consumption, less likelihood of error due to minimal human intervention, high throughput and sensitivity, automatic control, and low cost. With these advantages, DMFBs are gaining increasing applications, including DNA analysis, proteomic

Manuscript received June 14, 2012; revised August 27, 2012; accepted September 29, 2012. Date of current version January 18, 2013. The work of T.-Y. Ho was supported in part by the Taiwan National Science Council under Grant NSC 101-2220-E-006-016 and Grant NSC 101-2628-E-006-018-MY3. Preliminary version of this paper was presented at the 2012 ACM International Symposium on Physical Design. This paper was recommended by Associate Editor J. Hu.

J.-W. Chang, S.-H. Yeh, and T.-Y. Ho are with National Cheng Kung University, Tainan 701, Taiwan (e-mail: jwchang@eda.csie.ncku.edu.tw; hardyyeh@eda.csie.ncku.edu.tw; tyho@csie.ncku.edu.tw).

T.-W. Huang is with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78701 USA (e-mail: twhuang@eda.csie.ncku.edu.tw).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TCAD.2012.2224347

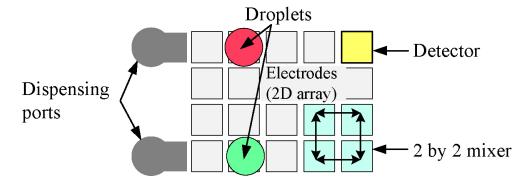


Fig. 1. Schematic view of a DMFB.

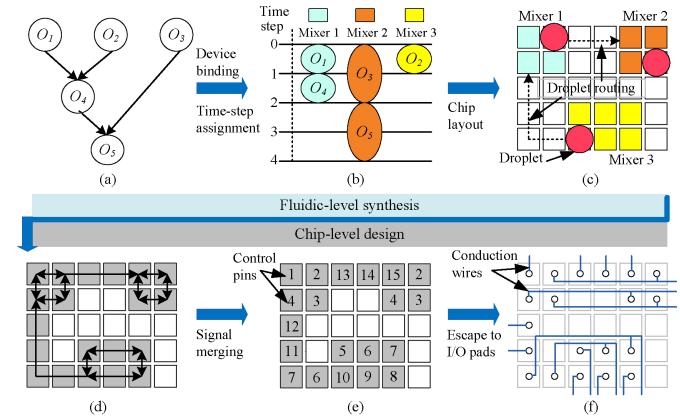


Fig. 2. Conventional design flow of DMFBs. (a) Sequencing graph. (b) Scheduling result. (c) Placement and routing. (d) Used electrodes. (e) Pin assignment. (f) Wiring solution.

analysis, immunoassay, and point-of-care diagnosis [1], [6], [8], [18], [20].

Generally, a DMFB consists of a 2-D electrode array and peripheral devices (optical detector, dispensing ports, etc.), as schematically shown in Fig. 1 [10], [20]. On a DMFB, the sample carriers (i.e., droplets) are controlled by underlying electrodes using electrical actuation (a principle called electrowetting-on-dielectric) [18]. Droplets are driven when nearby electrodes are applied with a control voltage. By assigning time-varying voltage values to turn on/off the electrodes, droplets can be moved around the entire 2-D array to perform fundamental operations. For instance, we can transport a droplet from a source location to a target location for a detecting operation, or cluster adjacent electrodes to form a mixing device. These operations are carried out in a reconfigurable manner due to their flexibility in area and time domain. That is, we can perform these operations anywhere on the 2-D plane during different time steps [22], [27].

As design complexity increases, computer-aided design (CAD) tools have been developed to build DMFBs efficiently.

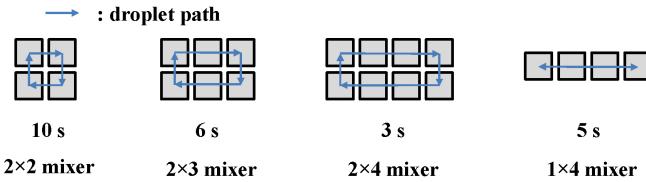


Fig. 3. Example of module library for mixing. For instance,  $2 \times 2$  mixer requires 10 s to execute.

Traditionally, designing and realizing DMFBs consist of two major stages, fluidic-level synthesis and chip-level design [10]. In fluidic-level synthesis illustrated in Fig. 2(a)–(c), different assay operations (e.g., mixing, dilution) and their mutual dependences are first represented as a sequencing graph. Each edge from  $o_a$  to  $o_b$  in the sequencing graph indicates that operation  $o_b$  must begin after operation  $o_a$  is finished. Next, scheduling and binding assigns time-multiplexed steps to these assay operations and binds them to a given number of devices so as to maximize parallelism [19]. In scheduling and binding, each operation will have a set of available devices to bind. For example, in Fig. 3, the type of mixing has different mixers to choose. Choosing different modules will result in various reaction area and execution time. With execution time, the beginning time and finishing time of each operation are arranged. Based on the scheduling result, device placement and droplet routing are conducted to generate a chip layout and establish droplet routing connections between devices in a reconfigurable manner [11], [21], [22]. The major goal in placement is to find the actual locations of different fluidic modules corresponding to different time intervals while in droplet routing is to construct the connections between modules within different time intervals. On the other hand, chip-level design, the second stage of conventional design flow, determines the required control pins and corresponding wiring connections for the underlying electrodes to execute the synthesis result. As illustrated in Fig. 2(d)–(f), fluidic-control information of used electrodes is obtained from the previous synthesis (i.e., fluidic-level synthesis). Then, control pins must be minimally and appropriately assigned to electrodes for minimizing the bandwidth of input signals. Several signal merging strategies are proposed to facilitate the pin assignment with minimum control pin usage [16], [24], [26], [29]. Finally, conduction wires must be routed to establish correspondence among control pins and external driving ports [12].

Regarding this conventional design flow, a number of high-quality CAD tools have been developed to solve several associated combinational optimization problems [11], [12], [16], [19], [21], [22], [24], [26], [27], [29]. Previous works regularly divide the entire DMFB designs into the fluidic-level synthesis followed by chip-level design. Due to the distinct nature between fluidic-level synthesis and chip-level design, most CAD tools are separately developed for the two design stages to simplify the complexity. For example, the works in [3], [4], [11], [19], [21], [22], [25], and [27] focus on specific stages in fluidic-level synthesis and the works in [3], [12], [24], and [26] deal with the chip-level design. Fig. 4 illustrates the statistics of previous works.

The works from top to down are arranged in the order of

Scheduling	Placement	Droplet routing	Pin assignment	Wiring
[19]				
	[22]	[21]	[24]	
	[27]	[25]		
		[4]	[26]	
		[11]		
		[16]		[12]
			[29]	
			[13]	
	[30]			
			[14]	
[9]				[3]

Fig. 4. Previous works in the design flow of DMFBs.

published year. It can be found that several works are proposed recently to simultaneously solve multiple phases so as to further optimize the DMFBs. However, existing fluidic-chip (fluidic-level and chip-level) design gap may still restrict the effectiveness and feasibility of the entire DMFB realization. Specifically, a successful fluidic-level-synthesis result cannot guarantee a successful chip-level-design solution. Fluidic-chip design gap is not concerned in previous works, and thus the feasibility to realize the entire DMFBs is restricted. Even though there is research [16], [29] that takes the pin count issue of chip-level design into consideration in an earlier stage, the wiring problem, which is a critical step for chip fabrication, is not referred to at all. Therefore, it may need additional processes such as pin reassignment, rerouting, extra pin-count demand, or even a multilayer routing structure to obtain a feasible wiring and successful chip-level design solution. Such processes add additional costs to chip fabrication and are not desirable for low-cost DMFBs. The fluidic-chip design gap problem will become even critical with a rapid escalation in the number of assay operations incorporated into a single large-scale DMFB. Unfortunately, to the best of our knowledge, there is still no work in the literature that provides a solution to deal with this concern.

#### A. Our Contributions

In this paper, we propose a fluidic-chip co-design methodology for DMFBs to effectively bridge the fluidic-chip design gap. Fig. 5 shows the comparison of conventional design flow and our co-design methodology. Compared with the conventional design flow, our co-design methodology is the first work in the literature that converges the fluidic-level synthesis and chip-level design. We provide a comprehensive integration throughout fluidic-operation scheduling, chip layout generation, control pin assignment, and wiring solution to achieve high design performance and feasibility. To realize the co-design methodology efficiently and effectively, we identify three major design concerns that must be solved to meet the design convergence.

- 1) Existing scheduling methods focus on parallel controls of assay operations to increase throughput. However,

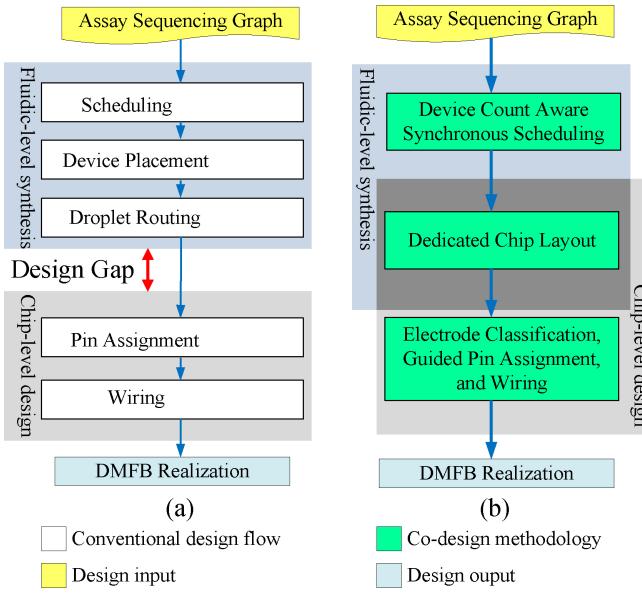


Fig. 5. Comparison of conventional design flow and proposed fluidic-chip co-design methodology.

device count is not restricted and thus much latter design effort is required especially for pin assignment and wiring.

- 2) Existing placement and routing methods allow devices and droplets to be arbitrarily moved in area and time domain. Although this scheme utilizes the reconfigurable properties, the entire solution incurs a lot of used electrodes with high pin-count demand and wiring problems.
- 3) Existing pin-assignment methods conduct the signal merging throughout the entire electrode set without careful arrangement. Despite achieving low pin count, the underlying wiring situation may confront congestion, detour, or even infeasible problems.

In order to handle these issues, our co-design methodology, as presented in Fig. 5(b), consists of three major stages.

- 1) Device count aware synchronous scheduling indeed reduces the required pin count and wiring effort.
- 2) Dedicated chip layout simultaneously determines the 2-D orientation of devices and routing paths, while simplifying and facilitating the designs of device placement, droplet routing, pin assignment, and wiring.
- 3) Electrode classification allows control pins to be orderly assigned with minimum wiring interference around different control signals. Guided pin assignment and wiring reduce the demanded pin count and wirelength for signal connections.

Along with the proposed co-design methodology, our contributions include the followings.

- 1) We propose the first fluidic-chip co-design methodology for DMFBs that provides an integration throughout fluidic-level synthesis and chip-level design and achieve high success rate of the entire DMFB realization.
- 2) For a given assay, we derive an exact integer linear programming (ILP) formulation to optimally minimize the device count that impacts on pin count and wiring

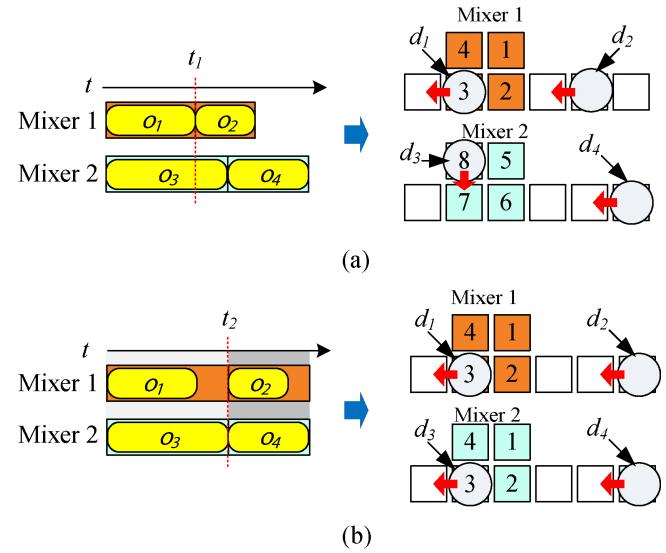


Fig. 6. Comparison of asynchronous reactions and synchronous reactions. The numbers of electrodes represent pin numbers assigned to electrodes. (a) Asynchronization. (b) Synchronization.

issue for chip-level design. An effective stage-count reduction scheme is also provided to reduce the problem size, while keeping assay completion time satisfied.

- 3) We propose a novel chip layout to avoid wiring congestion induced by arbitrary device placement and control signal merging in conventional design flow. With the proposed chip layout, following pin-assignment and wiring guidelines maintain the low pin-count and short wirelength.

Experimental results show the effectiveness and robustness of our co-design methodology for different real-life assays. Our co-design methodology successfully realizes all assays on DMFBs while the conventional design flow can only pass one case. We also explore how the success rate of DMFB realization changes when the problem size becomes large to demonstrate the scalability of the proposed co-design methodology. Evaluation shows that our co-design methodology always maintains high success rate, while the success rate realized by conventional design flow continues to decrease dramatically as the problem size increases.

The remainder of this paper is organized as follows. Section II details the concept and the proposed method of our methodology. Section III shows our experimental results compared with the conventional design flow. Section IV gives the conclusion.

## II. FLUIDIC-CHIP CO-DESIGN METHODOLOGY

In this section, we first describe the idea of synchronous reactions and then detail each phase of the proposed fluidic-chip co-design methodology for DMFBs.

### A. Synchronous Reactions

In pin-constrained digital microfluidic biochips [23], required pin count is a crucial issue as it directly influences the

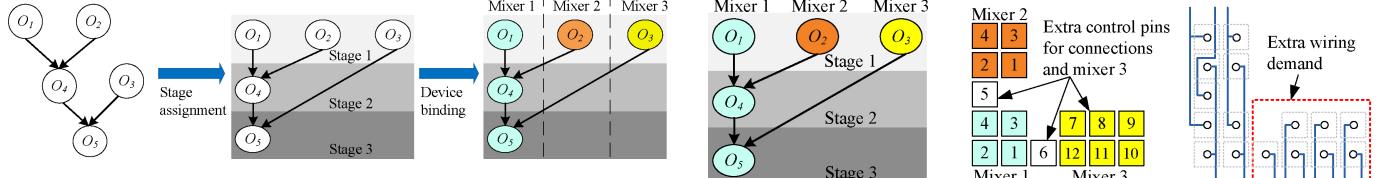


Fig. 7. Synchronous scheduling by stage assignment.

fabrication cost. Much research has been devoted to reducing the pin count for DMFBs. An effective manner is synchronous reactions. The objective of synchronous reactions is to facilitate the control signal merging for pin-count reduction by deriving a series of execution stages for synchronous controls. The differences between asynchronous and synchronous control can be explained by Fig. 6. Let us consider four operations  $o_1$ ,  $o_2$ ,  $o_3$ , and  $o_4$ , which are carried out by droplets  $d_1$ ,  $d_2$ ,  $d_3$ , and  $d_4$ , respectively. As shown in Fig. 6(a), asynchronous control methods in most previous works [19], [20] make each operation seamlessly enter into the subsequent operation in the end of its execution. Although this scheme maximizes the flexibility and independence of droplet controls, it may restrict the solution quality of pin-count reduction. Take the ending time step of  $o_1$  [ $t_1$  in Fig. 6(a)] for example, mixer 1 is ceased to let  $d_1$  leave, then followed by letting  $d_2$  in for  $o_2$ . Meanwhile, mixer 2 is still running for  $o_3$ , so  $d_4$  cannot enter into mixer 2 for  $o_4$ . To realize the scheduling result without any fluidic error, mixers 1 and 2 cannot share the same control pins. Thus, mixer 1 needs 4 control pins (pins 1, 2, 3, 4) and mixer 2 needs another four control pins (pins 5, 6, 7, 8). This design method may incur a high pin-count demand and is not suitable for low-cost DMFBs [16], [26]. In contrast, in Fig. 6(b), mixers 1 and 2 are controlled together, meaning that the mixers must begin and cease the executions synchronously. That is,  $o_1$  and  $o_3$  are simultaneously begun and ceased when the slower operation [ $o_3$  in Fig. 6(b)] is completed. Then,  $o_2$  and  $o_4$  simultaneously begin after  $o_1$  and  $o_3$  are both completed [ $t_2$  in Fig. 6(b)] and then ceased in the same feature. In this manner, mixers 1 and 2 can share their control signals [pins 1, 2, 3, 4 in Fig. 6(b)], which is more favorable for pin-count reduction.

To achieve synchronous reactions, the work in [16] expands the scheduling to a stage assignment problem by deriving a series of execution stages and align operations to these stages for synchronous controls. As an example shown in Fig. 7, to enable synchronous controls,  $o_1$ ,  $o_2$ , and  $o_3$  can be aligned to stage 1 and  $o_4$  and  $o_5$  can be aligned to stages 2 and 3, respectively. After binding all operations with appropriate devices, the synchronized control signals will simultaneously begin (cease) the execution in the start (end) of each stage. Then, Mixers 1, 2, and 3 can be controlled by the same set of control pins.

Although stage assignment provides a good solution for synchronous control, the followed chip-level design issue is not addressed. To realize the synchronous reaction, there must be conduction wires to connect synchronized devices together in wiring step. That is, introducing the synchronous reactions will inevitably incur wire congestion, detour, thereby increasing the complexity of wire routing. If device placement and droplet routing in fluidic-level synthesis are not carefully

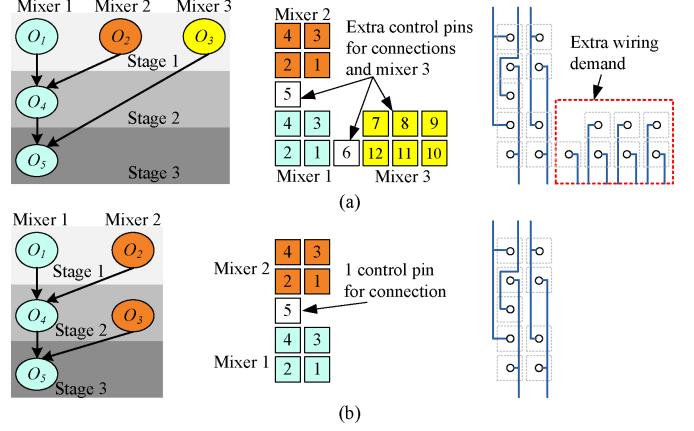


Fig. 8. (a) Scheduling result with three devices. (b) Scheduling result with two devices. Scheduling result with higher device count in (a) results in more control pins and wiring overhead than (b).

designed, resultant designs will be hardly realized in chip-level stage. As a result, a novel methodology to deal with this concern is needed. In following subsections, we propose a comprehensive co-design methodology to bridge the design gap between fluidic-level synthesis and chip-level design.

### B. Device Count Aware Synchronous Scheduling

Based on the concept of synchronous reactions, we observe that different stage-assignment results can lead to different device counts. As explained in Fig. 8,  $o_3$  can be aligned to two stages, stage 1 or stage 2, without violating the precedence on  $o_5$ . If we align  $o_3$  to stage 1 as shown in Fig. 8(a) additional mixing device, mixer 3, must be included for execution, implying extra pin count and wiring effort must be added to drive it. On the other hand, if we align  $o_3$  to stage 2 and then choose another mixing module as shown in Fig. 8(b)  $o_3$  can share the mixer 2 with  $o_2$  in different execution stages and thus the required number of devices can be decremented. In this case, we achieve a fewer number of control pins and simple wiring solution to realize the DMFB. As long as a device is placed on the microfluidic array, associated control pins and wiring must be included for this device and its connections to other devices. To avoid design overhead, it is consequently desirable to minimize the required device count for fluidic-operation scheduling.

1) *Problem Formulation:* Concerning those design issues discussed above, the scheduling problem can be formulated as follows.

*Given:* A sequencing graph of operations, a set of stages, and a device library.

*Objective:* Derive a set of execution stages for synchronous execution and bind all operations with appropriate devices to these stages, while minimizing the device count.

Note that the size of stage set is set as the total number of operations since the worst scheduling is executing one operation per stage.

2) *ILP Formulation:* To optimally solve the scheduling problem while minimizing the device count, we present an ILP formulation with the following notations, objective function, and constraints.

### Notations.

- 1) An operation set  $O$  with each operation indexed by  $i$ .
- 2) A device set  $D$  with each device indexed by  $j$ .
- 3) A stage set  $S$  with each stage indexed by  $k$ .
- 4) Available device set  $D_i$  for operation  $i$ ,  $\forall i \in O, D_i \subseteq D$ .
- 5) Execution time  $T_j$  for device  $j$ ,  $\forall j \in D$ .
- 6) A 0-1 variable  $o_{i,j,k}$  to represent that operation  $i$  is bound with device  $j$  and executed in stage  $k$ ,  $\forall i \in O, \forall j \in D_i, \forall k \in S$ .
- 7) An integer variable  $b_k$  ( $f_k$ ) to represent the beginning (finishing) time for stage  $k$ ,  $\forall k \in S$ .
- 8) An integer variable  $r_j$  to represent the required number for device  $j$ ,  $\forall j \in D$ .

*Objective:* The objective is to minimize the total device count as follows:

$$\text{Minimize} : \sum_{j \in D} r_j \quad (1)$$

*Subject to*

- 1) *Exclusivity constraints:* An operation is bound with one device and executed in one stage

$$\sum_{j \in D_i} \sum_{k \in S} o_{i,j,k} = 1 \quad \forall i \in O. \quad (2)$$

- 2) *Timing constraints:* For each stage, the beginning time should be less than the finishing time and all stages should be finished by a maximum timing value

$$0 \leq b_k \leq f_k \leq T_{max} \quad \forall k \in S. \quad (3)$$

Note that  $T_{max}$  can be set to an upper bound of the minimum assay completion time obtained by greedy assignment or a user-defined value [19].

- 3) *Synchronization constraints:* In realizing the synchronization, all stages must be orderly executed without overlapping

$$f_{k_1} \leq b_{k_2} \quad \forall k_1 \in S \quad \forall k_2 \in S, k_1 < k_2. \quad (4)$$

Besides, a stage will not enter into the subsequent stage until all included operations have been finished. In other words, the duration of a stage is lower-bound by the slowest operation assigned to the stage

$$f_k - b_k - T_j o_{i,j,k} \geq 0 \quad \forall k \in S \quad \forall i \in O \quad \forall j \in D_i. \quad (5)$$

- 4) *Dependency constraints:* If there is a dependency “ $\rightarrow$ ” between operations  $i_1$  and  $i_2$  (i.e., operation  $i_2$  should be executed after operation  $i_1$  is finished), the stage  $k_2$  that includes operation  $i_2$  must appear after the stage  $k_1$  that includes operation  $i_1$

$$f_{k_1} - b_{k_2} + M(o_{i_1,j_1,k_1} + o_{i_2,j_2,k_2} - 2) \leq 0 \\ \forall i_1 \rightarrow i_2, j_1 \in D_{i_1}, j_2 \in D_{i_2}, k_1 \in S, k_2 \in S. \quad (6)$$

Note that  $M$  is a big number and must be larger than  $T_{max}$  for correct formulations.

- 5) *Device requirement:* The inherent reconfigurability of DMFBs allows operations to share the same device in

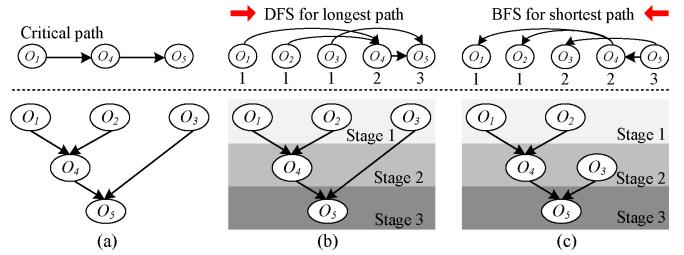


Fig. 9. Finding the stage intervals (earliest and latest available stages with respect to the critical path) for operations to reduce the ILP complexity. (a) Sequencing graph. (b) Earliest available stage. (c) Latest available stage.

different execution stages. Therefore, the required number for each device should be more than its requirement in any stage

$$r_j \geq \sum_{i \in O} o_{i,j,k} \quad \forall j \in D \quad \forall k \in S. \quad (7)$$

Note that the summation of  $o_{i,j,k}$  is only calculated when  $j \in D_i$  to avoid the boundary overflow.

3) *Solution Space Reduction:* Although the basic ILP formulation presented in the previous subsection can optimally solve the scheduling problem, it is complicated for large-scale assays and not efficient to be solved without reductions. The major difficulty is the number of execution stages, which dominates the major complexity of constraints and variables. Thus, the stage count must be conditionally bound to avoid runtime overhead.

To handle such an issue, we conduct explicit stage assignments for all operations. By observing the sequencing graph, it can be found that the available stages of operation can be bound and restricted. Therefore, we assign a stage interval to each operation  $o_i$ , denoted as a set  $S_i$  of a consecutive stage sequence, to make operation  $o_i$  only executed in these stages instead of all stages. The objective is to minimally restrain the sizes of stage intervals while considering the dependencies around operations. Since the sequencing graph is a directed acyclic graph, the number of stages can be bound by the length of a critical path [10], [19], [20]. With respect to the critical path, we can easily obtain the stage interval (i.e., earliest and latest available stages) in which an operation should be executed.

In order to keep the quality of solution space after reduction, we calculate the earliest and latest available stage for each operation by depth-first search (DFS) and followed by breadth-first search (BFS). For example, suppose we have a sequencing graph with five operations, as shown in Fig. 9(a). With respect to the critical path,  $o_1 \rightarrow o_4 \rightarrow o_5$ , we conduct DFS to trace the longest paths from beginning nodes  $\{o_1, o_2, o_3\}$  (i.e., zero indegree) to all other nodes, as shown in Fig. 9(b). Note that the distance values of beginning nodes are initialized as 1. So now we can know the earliest stage in which  $o_1, o_2, o_3, o_4, o_5$  can be executed are 1, 1, 1, 2, 3 [as shown in (b)]. Then, we reverse the sequencing graph and conduct BFS to backtrace the shortest paths from the ending node  $\{o_5\}$  (i.e., zero outdegree) to all other nodes. Note that the distances of ending nodes are initialized as their longest distance values and

will decrease level by level in BFS routine. This way, the latest available stages for each operation can be obtained. Since all executions should follow the dependencies, the longest (shortest) distances thus represent the earliest (latest) stages in which operations can be executed. The advantages of reduction are obvious. For example, the available stages for operation  $o_3$  are all stages (i.e., 1, 2, 3, 4, 5) before. But it becomes stage 1 to stage 2 after reduction, which eliminates three candidates after reduction.

After finding all stage intervals  $S_i$ , we replace  $S$  with  $S_i$  for (2), (5), and (6) in our ILP formulation. Since  $S_i$  is relatively small compared with  $S$ , the entire solution space and complexity of constraints and variables can be significantly reduced. Besides, the proposed reduction method follows the dependencies when assigning stage intervals for operations. The dependency constraints around those operations with only one assigned stage become redundant and thus can be removed. Consider Fig. 9(b) and (c) for example, since both earliest and latest stages of operations  $o_1$  ( $o_4$ ) are assigned by stage 1 (stage 2), the dependency constraint for  $o_1 \rightarrow o_4$  can be removed.

The advantages of the proposed reduction methods are threefold. First, since the stage count is tightly bound by the length of the critical path, the assay completion time can be also minimally bound without excessively suspending operations between stages. On the other hand, minimally assigning the stage intervals can also avoid redundant solution search and restrict unnecessary formulations of constraints and variables. Furthermore, our reduction method also provides flexibility. If the initial assignments are too tight, we can easily and incrementally expand the stage intervals to find a better solution without numerous modification of the entire routine. That is, we can expand all latest available stages for all operations by a user-defined range. Since such expansion may result in longer assay completion time and ILP solving time, we empirically set the expansion to be 1.

### C. Schedule Refinement

After deciding the required device, we would like to minimize the completion time of bioassays. In the formulation of Section II-B, the assay completion time is not minimized. Such formulation may result in a longer completion time instead of a shorter one with the same device count. As explained in Fig. 10, scheduling in Fig. 10(a) completes at 30 s. Duration of Stage 2 is bound and dominated by operation  $o_3$  since it needs a relative long time to execute. This situation occurs on  $o_1$  similarly in stage 1. Obviously, some modifications can be made to reduce the completion time. If we move  $o_3$  from stage 2 to stage 1, the duration of stage 2 will decrease. The modified result is shown in Fig. 10(b), which brings a shorter completion time, 27 s. A minimized assay completion time is essential because many applications especially for surgery and clinical diagnostics are timing sensitive. Although the completion time is bound with the maximum timing value user define, it still needs to be minimized to reduce the likelihood of unexpected error such as functional error or catastrophic defect when assays are executed. A way to achieve such goal is to integrate this concern into the objective function

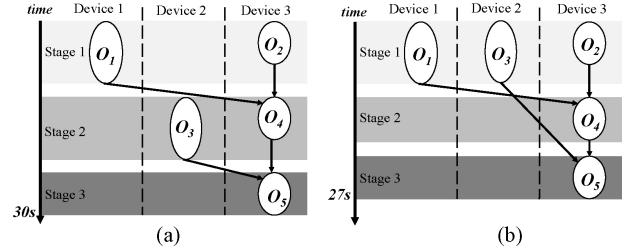


Fig. 10. Adjustment of operations among stages can reduce the assay completion time without adding an extra device.

of ILP formulation and then set a higher priority to device count. However, this will increase the complexity and the time to solve ILP. Thus, a simple method to quickly reduce the completion time within the same device count is required.

To this end, we apply a heuristic strategy as the following steps. The main idea is to shorten the duration time of each stage by moving away the operations that dominate stage duration time.

- 1) Given the stage assignment obtained from ILP result, each stage contains a set of operations assigned to this stage.
- 2) Randomly pick a stage from the stage set.
- 3) Collect critical operations from this stage. Critical operations are the operations that dominate the duration of this stage.  $o_3$  in Fig. 10 is an example of critical operation. In this example, if we remove this operation, the duration of this stage can be reduced. Note that several operations may exist that dominate duration time.
- 4) Move these operations to other available stages without increasing duration time of other stages and violating the operation dependency.
- 5) If critical operations are moved away from this stage, duration time of this stage is reduced.
- 6) Repeat steps 2–5 until the whole assay completion time can no longer be reduced.

With this post-refinement, we can obtain a shorter assay completion time for real-time response assay demand.

### D. Chip Layout Construction

After all operations are scheduled with minimum device count, these devices must be well placed to generate a chip layout. Most previous device placement methods focus on determining the device locations during different time steps to guarantee correct execution and feasible routing solution. Strategies such as segregation between devices to avoid unexpected mixing and dynamically moving devices to free more cells for routing are also considered [10], [22], [27].

Although these works can solve the device placement and droplet routing, they may suffer from chip-level design problems. An essential problem of previous device placement methods is the excessive allowance of a cell (i.e., electrode) to be arbitrarily functioned, such as mixing, diluting and detecting devices in different time steps. However, different devices are always associated with different control signals, implying that such methods require high pin-count demand to maximize

Pin	Cell	time		
1	$C_1$	Mixer	Dilutor	Mixer
2	$C_2$	Mixer	Dilutor	Mixer
3	$C_3$	Mixer	Mixer	Dilutor
4	$C_4$	Dilutor	Mixer	Dilutor
5	$C_5$	Dilutor	Mixer	Mixer

Pin	Cell	time		
1	$C_1$	Mixer	Mixer	Mixer
1	$C_2$	Mixer	Mixer	Mixer
1	$C_3$	Mixer	Mixer	Mixer
2	$C_4$	Dilutor	Dilutor	Dilutor
2	$C_5$	Dilutor	Dilutor	Dilutor

(a)

(b)

Fig. 11. (a) Cells are arbitrarily functioned with a pin count of 5. (b) Cells are dedicated to specific functions with a pin count of 2.

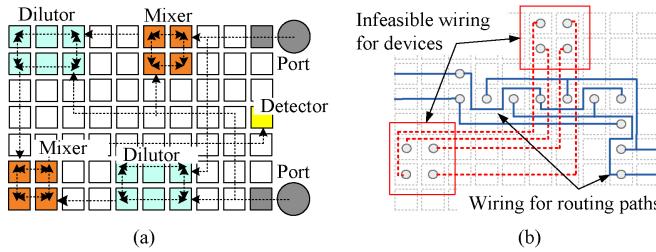


Fig. 12. Spreading out the droplet routing paths around devices may lead to wiring problems. (a) Droplet routing solution. (b) Fragment of wiring solution.

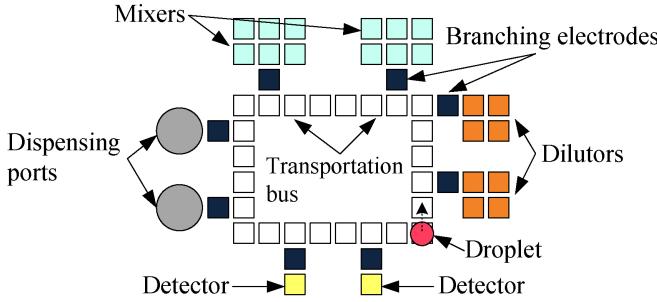


Fig. 13. Proposed chip layout.

the signal bandwidth and freedom of electrode controls, as shown in Fig. 11(a). Thus, to improve this deficiency, it is desirable to make cells dedicated to unique devices or specific functions. As shown in Fig. 11(b), by dedicating cells to specific functions, cells with the same action can be controlled synchronously. What we concern now is determining the fixed 2-D location on specific cells for each device, while achieving feasible droplet routing solutions.

On the other hand, the major problem of previous droplet routing methods is that they spread out the routing paths between devices for establishing connections [11], [21], as illustrated in Fig. 12(a). Although such methods utilize the temporal flexibility of DMFBs, the induced wiring complexity may trigger many blocking or detouring problems. Consider Fig. 12(b) for example, the wiring solution for droplet routing paths significantly blocks the wiring for devices, which makes an infeasible design outcome especially for connecting synchronous devices. Hence, it is necessary to conduct a specific orientation for routing paths with device connections, while taking the chip-level design issues into consideration.

To deal with these issues, the device placement and droplet routing for the entire assay should be decided simultaneously on a 2-D plane. In this concern, we propose a dedicated chip layout to deal with these issues and support our scheduling method. As presented in Fig. 13, the architectural layout

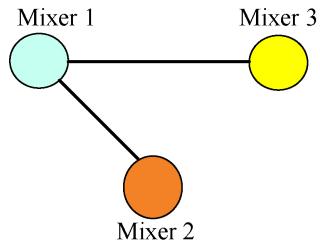


Fig. 14. Illustration of connection graph.

consists of three major components, surrounding devices for executing operations, central transportation bus for droplet routing and device connections, and branching electrodes for accessing the passages between bus and devices. With the proposed chip layout, we achieve three major advantages as follows.

- 1) Since the outside surrounding devices are fixed without arbitrary movements and the inside bus is independently oriented without significant interference with devices, the aforementioned wiring problem can be avoided.
- 2) In addition to routing droplets, the central transportation bus can serve as storages for intermediate droplets. Thus, extra storing devices for holding droplets are omitted and thus the design effort can be reduced.
- 3) Instead of directly constructing the chip layout in 3-D configuration ( $z$ -axis for time), the proposed layout focuses on determining the 2-D orientation of devices and transportation bus, which greatly reduces the design complexity.

Regarding this, different device orderings and bus lengths may produce different droplet routing distances and realizations of the previous scheduling results. Consider the sequencing graph in Fig. 8(a). Each edge in the sequencing graph implies the droplet transportation if the incident two operations are bound with different devices. We refer to this droplet transportation as connection between these devices. Take the edge from  $o_3$  to  $o_5$  as an example. If  $o_3$  is bound with mixer 3 and  $o_5$  is bound with mixer 1, it then forms a connection between mixer 1 and mixer 3 because droplets must be transported from mixer 1 to mixer 3 after  $o_3$  is finished. Fig. 14 shows the connection graph derived from Fig. 8(a). The connections can be used to represent the relationships between devices and to evaluate the droplet transportation length. For instance, we can get a shorter transportation length if we put the device of  $o_3$  and  $o_5$  closer. That is, a means to determine the ordering of surrounding devices and the length of transportation bus is the major concern of constructing the chip layout. Note that the central transportation bus is bidirectional. We allow the routing direction of droplets on the central transportation bus can be clockwise or anticlockwise. Hence, the droplet will choose a shortest path to its target device by calculating each moving direction. The droplet will move clockwise if it is close to the target in the clockwise direction, and vice versa.

- 1) *Problem Formulation:* Regarding those issues discussed in the previous subsection, the problem of constructing a chip layout can be formulated as follows.

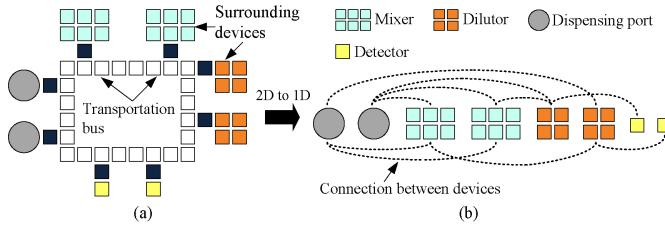


Fig. 15. Example of transforming surrounding devices to an 1-D ordering. (a) Chip layout. (b) 1-D device ordering.

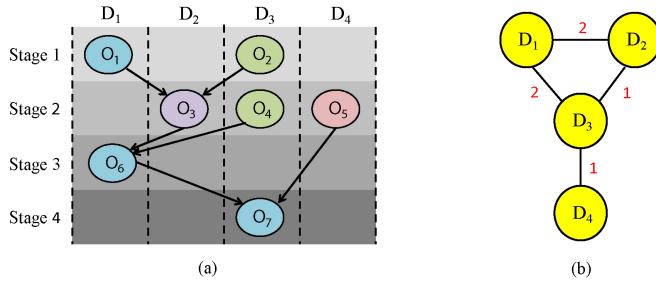


Fig. 16. Example of connection graph and its corresponding sequencing graph. (a) Sequencing graph. (b) Connection graph.

*Given:* A set of devices and their connection relations.

*Objective:* Derive a device ordering to minimize the total transportation distances for droplets, while minimally determining the bus length to supply all device connections and required droplets.

2) *Linear Assignment for Device Ordering:* As droplets are routed between devices, if device orderings are not well-oriented, a longer droplet routing time is required, which may potentially affect the assay throughput [11], [20], [21], [29]. Since it is not straightforward to recognize the device ordering on a 2-D plane, we transform the surrounding devices to an 1-D ordering, as demonstrated in Fig. 15. Referring to the scheduling result, we can obtain the connections between devices due to original operation dependencies. A simple method for solving the problem is to assign adjacent devices with one unit distance. The problem now is deriving an 1-D ordering that we need to minimize the total induced distances of devices connections.

The problem that we formulate above is actually a classic linear assignment problem, which is well known as NP complete [7]. This feature justifies the necessity of using heuristic approaches to solve our problem. Fortunately, there are high-quality heuristics and approximation algorithms for solving the linear assignment problem. In this paper, a typical method is deriving by iteratively swapping pairs of the devices as long as the total distances can be reduced. The swaps are repeatedly conducted until an equilibrium is reached.

3) *Explicit Estimation for Linear Assignment:* As we mentioned above, a typical heuristic method is conducted to solve the linear assignment problem, while the solution represents the device ordering in the chip's layout. However, the transportation time of droplets can be reduced since the model of connection graph is not explicit enough. An explicit estimation model for connection graph is introduced

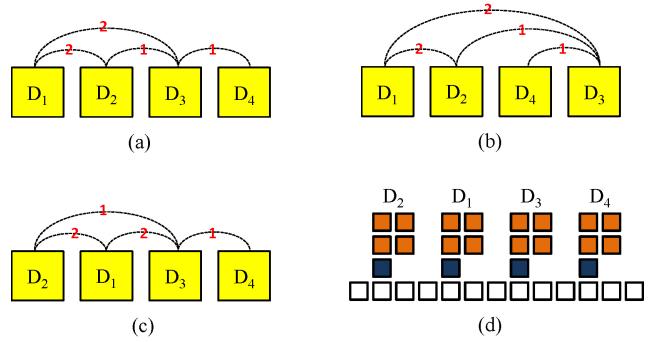


Fig. 17. Three solutions of linear assignment with (a) cost = 8, (b) cost = 11, and (c) cost = 7. (d) Result of device order on the chip layout.

as follows. In Fig. 16(a), there are four devices ( $D_1 \sim D_4$ ) and their corresponding sequencing graph [Fig. 16(b)]. Every connection between devices is considered equal in the connection graph in the previous model. Now, we not only assign connections between devices, but also count the number of operation relationships between devices. The connection graph becomes a weighted graph, while the weight of an edge represents the number of operation relationships between devices in the sequencing graph. Consider Fig. 16(a) for example, there are two connections (( $O_1, O_3$ ) and ( $O_3, O_6$ )) between  $D_1$  and  $D_2$ . Thus, the weight of edge between  $D_1$  and  $D_2$  in the connection graph is 2. Similarly, the weight of edge between  $D_1$  and  $D_3$  is 2 due to the two connections of ( $O_4, O_6$ ) and ( $O_6, O_7$ ). All the operation relationships in a sequencing graph must be considered in a connection graph.

After the weighted connection graph was constructed, a cost can be computed for each solution of linear assignment. Given a candidate solution of linear assignment, we first calculated the distances between every pair of devices. (We assumed that the distance between adjacent devices is 1, e.g., 1 for adjacent devices, 2 for devices with one occupied in the middle.) Then, in a connection graph, we calculated the products of each edge's weight and its corresponding distance between devices. Finally, cost can be obtained by summing all the products up. The cost function is shown in (8). We use  $Dis(D_i, D_j)$  to represent the distance between  $D_i$  and  $D_j$ ,  $G_c$  to represent connection graph

$$Cost = \sum_{e \in G_c} W_e \times Dis(D_i, D_j), (\{D_i, D_j\} \in e). \quad (8)$$

Fig. 17(a)–(c) shows a example of cost calculation. It contains three candidate results of linear assignment corresponding to the connection graph in Fig. 16(b). For Fig. 17(a), the costs caused by four edges are  $2 \times 2, 2 \times 1, 1 \times 1, 1 \times 1$  and total cost would be  $2 \times 2 + 2 \times 1 + 1 \times 1 + 1 \times 1 = 8$ . By swapping  $D_3$  and  $D_4$  of Fig. 17(a), we get the result of Fig. 17(b) whose cost is 11 ( $2 \times 3 + 2 \times 1 + 1 \times 2 + 1 \times 1 = 11$ ). A better result [Fig. 17(c)] can be obtained by swapping  $D_1$  and  $D_2$  of Fig. 17(a) with cost equal to 7 ( $1 \times 2 + 2 \times 1 + 2 \times 1 + 1 \times 1 = 7$ ). Notice that with this model the total time of droplet routing can be decreased by reducing the distance of droplet transportation. Fig. 17(a) and (c) is seen to be the same in the previous method that is not explicit. Assume that we take the candidate

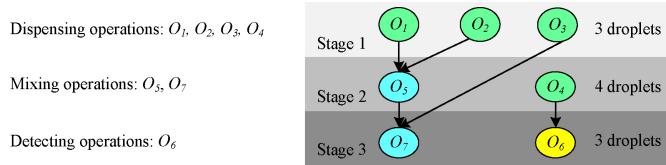


Fig. 18. Determining the bus length to supply enough droplets.

device order in Fig. 17(c) to be our desired result. The devices on the chip layout can be represented by Fig. 17(d).  $D_1-D_4$  are arranged on the chip layout according to the result from Fig. 17(c). Notice that all the devices of the same kind would be operated synchronously and arranged together. As we mentioned in Section II-A, they should be assigned with the same pins and arranged together since there must be conduction wires to connect synchronized devices together in a wiring step.

Besides the minimization of the droplet routing time, chip-level design issues must also be considered. An essential concern is the devices with the same type (e.g., mixing devices or detecting devices) that are synchronized and controlled together. These synchronized devices share the same control signals to reduce the pin count. Considering the wiring issues, it is desirable to place these devices closely so as to minimize the wirelength and to simplify the wiring problem. Therefore, for synchronized devices, we contract them as a single device in formulating the linear assignment problem to guarantee that they are placed in adjacent positions.

4) *Transportation Bus Length Determination:* Another key issue is to determine enough length of transportation bus to supply droplets for realizing the scheduling result. In other words, following pre-assigned execution stages, the bus length must be minimally bounded while accommodating the maximum number of droplets that are delivered between stages.

The droplet accommodation of the bus can be determined by examining the droplet requirement for each stage while picking a maximum one as the solution. For each stage, the droplet requirement comes from two parts, its own dispensing operations and predecessor operations. As explained by Fig. 18, since there are only three dispensing operations in stage 1, the bus must supply three droplets. In stage 2, there are one dispensing operation  $o_4$  and three predecessor operations,  $o_1, o_2$ , and  $o_3$  with edges passing through this stage, and thus we need four droplets on the bus for this stage. In the same way, stage 3 has only three predecessor operations,  $o_3, o_4$ , and  $o_5$  that will deliver droplets to this stage, and thus we need three droplets on the bus for this stage. In these concerns, a transportation bus that can supply simultaneous movements of at least four droplets is required. To avoid unexpected mixing between droplets, a minimum segregation of four cells between two droplets must be included (we will discuss it in the next subsection). Therefore, the bus length can be determined as a multiple of 4. That is, total  $4 \times 4 = 16$  cells (i.e., electrodes) are required for constructing the transportation bus for this example. Then, based on predetermined device ordering, we can circularly connect devices to the bus by adding branching



Fig. 19. Synchronized droplet movements require at least four control pins to prevent the fluidic error. (a) Three pins for droplet routing. (b) Four pins for droplet routing.

cells to generate the entire 2-D chip layout. Note that there are three detailed implementation concerns as follows:

- 1) both square and rectangular orientations of the bus are allowed;
- 2) segregation between devices is also included to avoid unexpected mixing;
- 3) if devices are too large to be connected by current bus length, we incrementally increase the bus length by 4.

#### E. Pin Assignment and Wiring Solution

To realize the scheduled fluidic functions, control pins must be appropriately and minimally assigned on the chip for low-cost fabrication [24], [26]. Besides synchronizing the devices in each stage, we also synchronously control the droplets to achieve pin sharing on the transportation bus. A primary issue is the avoidance of fluidic errors between droplets during control signal sharing. As explained in Fig. 19, to avoid undesirable fluidic errors (e.g., mixing or splitting) caused by neighboring actuations, we require at least four control pins on the transportation bus such that a safe spacing can be maintained between droplet movements [21].

After control pins are assigned on the chip, wires must be routed to establish the correspondence between control pins and then escape to outside I/O pads. Since multiple electrodes may share the same control pins, the wiring problem thus can be viewed as an escape routing with multi-terminal pins. However, such a wiring problem is well known as NP-complete and is computationally expensive if we directly solve it [10], [12]. Therefore, a specific method must be developed to handle this issue.

- 1) *Problem Formulation:* The pin-assignment and wiring problems can be formulated as follows.

*Given:* A scheduling result and the corresponding chip layout.  
*Objective:* Derive a pin-assignment result and establish wiring connections between control pins to realize the fluidic-level synthesis result without introducing erroneous fluidic behaviors.

- 2) *Classifications of Pins and Wires:* The major reason that the previous pin-assignment method suffers from wiring problems is the excessive signal merging without any restriction [12], [26], [29]. Since the pins with the same control signal must be wired together to receive the same input signal, overly merging the control signals around all electrodes may potentially incur detours, blocking, or even deadlock between wires. Therefore, it is desirable to make a specialized plan throughout the entire electrode set to avoid this problem.

To this end, we classify all electrodes to three categories, bus, branch, and device by their functionality and conduct pin

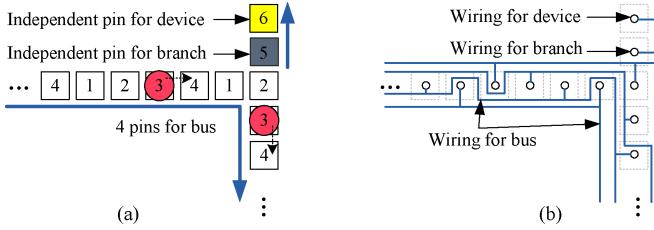


Fig. 20. Pin assignment and wiring for transportation bus, branch, and device. (a) Pin assignment. (b) Wiring.

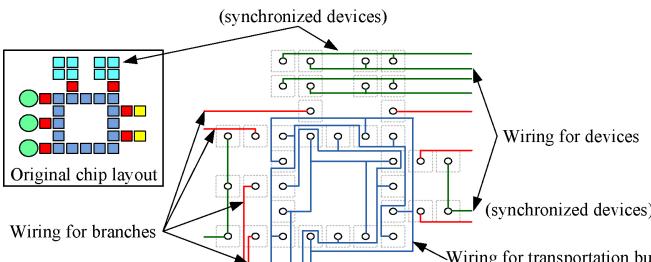


Fig. 21. Realization of wiring solution.

assignment and wiring individually for the three categories, as explained in Fig. 20. Regarding the pin-constrained design issue [10], [20], [26], we propose the following pin-assignment guidelines for pin-count reduction.

- 1) Since droplets are synchronously moved, four independent control pins 1, 2, 3, and 4 are sequentially and repeatedly assigned to the transportation bus.
- 2) Branching electrodes are assigned by dedicated control pins for independent accesses of the passages between transportation bus and devices.
- 3) Devices with the same type are synchronously controlled and shared the same control pins.

As we comprehensively address the wiring issues in the proposed co-design methodology, wiring connections between control pins can be systematically established based on the following guidelines.

- 1) Since there are always four control pins assigned on the well-oriented transportation bus, the associated wiring solution can be sequentially constructed from innermost region to outermost region, following the pin order 1, 2, 3, 4 (see the wiring for transportation bus in Fig. 21).
- 2) Since synchronized devices are placed adjacent to each other, shared control pins can be wired together without lots of detours so as to minimize the wirelength (see the wiring for devices in Fig. 21).
- 3) For branching electrodes that are assigned by dedicated control pins, the preserved segregation between devices provides region for wiring these pins (see the wiring for branches in Fig. 21).

Note that since we have minimized the required device count in previous scheduling, the transportation bus length and branching electrodes can also be minimally bound. Thus, the entire pin-assignment and wiring effort is also reduced.

TABLE I  
COMPARISON BETWEEN THE CONVENTIONAL DESIGN FLOW AND OUR FLUIDIC-CHIP CO-DESIGN METHODOLOGY

Assay	Conventional design				Ours				
	#D	T	#P	#WL	#D	T'	#P	#WL	
Multiplexed	7	63	25	N/A	6	67	55	14	562
DNA	11	67	38	1094	11	79	69	22	610
Diagnostic	11	50	26	N/A	9	63	60	18	748
PCR	14	26	14	N/A	10	47	47	18	820
Total	43	206	95		34	256	231	72	

#T: assay completion time in [15].

#T': assay completion time after conducting schedule refinement and explicit estimation (s).

#D: device count. #P: pin-count.

#WL: wirelength (measured by the number of used grids).

### III. EXPERIMENTAL RESULTS

We implement our fluidic-chip co-design methodology in C++ language on a 2 GHz 64-b Linux machine with 16 GB memory and use GLPK [2] as our ILP solver. Evaluations are based on a set of real-life chip applications for multiplexed assay, DNA sequencing assay, in-vitro diagnostic assay, and PCR amplification assay [17], [19], [26]. We use the real chip specification of the wiring model in [12]. To show the effectiveness of our integrated design method, we implement the conventional design flow for DMFB realization by five stages of scheduling, device placement, droplet routing, pin assignment, followed by wiring. For the purpose of comparison, we choose four state-of-the-art works from the same group at Duke University as our conventional design flow algorithms. The works in [19], [21], [22], and [26] were conducted as the solvers for scheduling, device placement, droplet routing, and pin assignment, respectively. Then, since the associated wiring problem is NP-complete, we conduct the approach in [12] that is based on maze routing to sequentially and iteratively route a nearest electrode pair with the same control pin until all electrodes are routed.

As listed in Table I, the overall comparison results show that our integrated fluidic-chip co-design methodology successfully accomplishes all assays by correctly deriving the result from fluidic-level synthesis to chip-level design. However, the conventional design flow can only accomplish one assay of DNA sequencing. The major reason is that the conventional design flow requires large numbers of pin-count and all the unsuccessful assays are running into infeasible wiring solution since conventional design did not integrate the relationship between pin-assignment and wiring connection. That is why we intend to bridge this gap. In our synchronous scheduling, we first minimize the required device count by 21% for reducing the effort in later design, especially in chip level. The increase of assay completion time is expected, since synchronization may delay some operations in an execution stage until the slowest one is finished. With the improvement of schedule refinement and explicit estimation in this paper, the completion time can effectively be reduced by 20%. Comparison to [15] was shown in Table I. Based on the scheduling result, we

TABLE II  
RUN TIME RESULTS OF THE OPTIMAL ILP FORMULATION AND REDUCED  
ILP FORMULATION

Assay	Optimal ILP		Reduced ILP		
	#D	CPU	#D	Error	CPU
Multiplexed	6	> 1 day	6	0%	0.04
DNA	11	> 1 day	11	0%	0.07
Diagnostic	7	> 1 day	9	29%	0.12
PCR	9	> 1 day	10	11%	0.10

#D: device count CPU: cpu run time in seconds

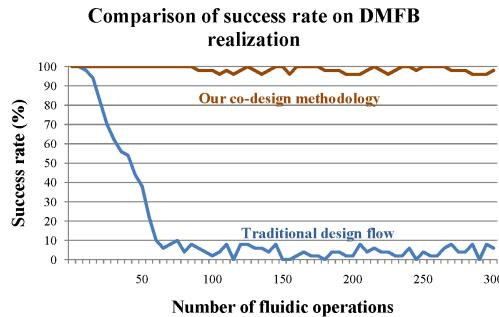


Fig. 22. Comparison of the success rate on DMFB realization between our co-design methodology and conventional design flow.

propose a dedicated chip layout for realizing the scheduling result while considering the impact on chip-level designs. With appropriate electrode classification to avoid the wiring congestion followed by pin-assignment and wiring guidelines, we reduce the required pin count by 30%, while maintaining successful wiring solutions for all assays. Even compared with the only DNA sequencing assay that can successfully be realized by the conventional design flow, we also reduce the wirelength by 44%. Note that in CPU runtime all assays can be solved within 1 s by conventional design flow and our co-design methodology.

In runtime evaluation, solving the ILP formulation for synchronous scheduling with minimum device count dominates the entire runtime performance by about 90%–95%. We list the runtime results of the optimal ILP formulation and the proposed reduced ILP formulation in Table II. Compared with the optimal ILP formulation, our reduced ILP formulation can efficiently solve all the assays in a much more reasonable CPU runtime, all by less than 1 s. We also achieve two optimal results in multiplexed and DNA assays.

Then, we randomly generate a set of assays with large problem size to demonstrate the scalability of our co-design methodology. The problem size is evaluated as the number of operations in an assay. For each problem size, we simulate the fluidic operations and generate 50 assays to test the success rate of DMFB realization (from fluidic-level synthesis to chip-level design without any failure) of our co-design methodology and conventional design flow. As illustrated in Fig. 22, our co-design methodology always maintains high success rate of average 99% for all test assays (note that our fluidic-chip co-design methodology can efficiently solve the largest assay with

300 operations by taking only 23 s). Nevertheless, the success rate by conventional design flow decreases dramatically as the problem size increases. We find out that when the problem size increases to about 60 operations, the success rate is always lower than 10%, which is not strong enough to realize many large-scale DMFBs. This justifies the deficiency of conventional that ignores the design gap between stages. Besides, our fluidic-chip co-design methodology can efficiently solve the largest assay with 300 operations by taking only 23 s.

#### IV. CONCLUSION

In this paper, we presented the first integrated fluidic-level and chip-level co-design methodology for DMFBs that provided an integration throughout fluidic-level synthesis and chip-level design. We also comprehensively identified the factors that would affect DMFB realization and explore properties that are favorable for bridging the fluidic-chip performance gap. Experimental results demonstrated that our co-design methodology achieved higher success rate of DMFB realization over the conventional design flow.

#### REFERENCES

- [1] Advanced Liquid Logic, Inc. [Online]. Available: <http://www.liquid-logic.com>
- [2] GNU Linear Programming, Kit, Free Software Foundation, Inc. [Online]. Available: <http://www.gnu.org/software/glpk/>
- [3] J.-W. Chang, T.-W. Huang, and T.-Y. Ho, “An ILP-based obstacle-avoiding routing algorithm for pin-constraint EWOD chips,” in *Proc. IEEE/ACM ASP-DAC*, Jan. 2012, pp. 67–72.
- [4] M. Cho and D. Z. Pan, “A high-performance droplet-routing algorithm for digital microfluidic biochips,” *IEEE Trans. Comput.-Aided Design*, vol. 27, no. 10, pp. 1714–1724, Oct. 2008.
- [5] K. Chakrabarty, and F. Su, *Digital Microfluidic Biochips: Synthesis, Testing, and Reconfiguration Technique*. Boca Raton, FL: CRC Press, 2006.
- [6] R. B. Fair, “Digital microfluidics: Is a true lab-on-a-chip possible,” *Microfluid. Nanofluid.*, vol. 3, no. 3, pp. 245–281, 2007.
- [7] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman, 1979.
- [8] J. Gong, S.-K. Fan, and C.-J. Kim, “Portable digital microfluidic platform with active but disposable lab-on-chip,” in *Proc. IEEE MEMS*, Jan. 2004, pp. 355–358.
- [9] D. Grissom and P. Brisk, “Path scheduling on digital microfluidic biochips,” in *Proc. ACM/IEEE DAC*, Jun. 2012, pp. 26–35.
- [10] T.-Y. Ho, J. Zeng, and K. Chakrabarty, “Digital microfluidic biochips: A vision for functional diversity and more than Moore,” in *Proc. IEEE/ACM ICCAD*, Nov. 2010, pp. 578–585.
- [11] T.-W. Huang and T.-Y. Ho, “A fast routability- and performance-driven droplet routing algorithm for digital microfluidic biochips,” in *Proc. IEEE ICCT*, Oct. 2009, pp. 445–450.
- [12] T.-W. Huang, S.-Y. Yeh, and T.-Y. Ho, “A network-flow based pin-count aware routing algorithm for broadcast-addressing EWOD chips,” *IEEE Trans. Comput.-Aided Design*, vol. 30, no. 12, pp. 1786–1799, Dec. 2011.
- [13] T.-W. Huang, S.-Y. Yeh, and T.-Y. Ho, “A two-stage ILP-based droplet routing algorithm for pin-constrained digital microfluidic biochips,” *IEEE Trans. Comput.-Aided Design*, vol. 30, no. 2, pp. 215–228, Feb. 2011.
- [14] T.-W. Huang, K. Chakrabarty, and T.-Y. Ho, “Reliability-oriented broadcast electrode-addressing for pin-constrained digital microfluidic biochips,” in *Proc. IEEE/ACM ICCAD*, Nov. 2011, pp. 448–455.
- [15] T.-W. Huang, J.-W. Chang, and T.-Y. Ho, “Integrated fluidic-chip co-design methodology for digital microfluidic biochips,” in *Proc. ACM ISPD*, 2012, pp. 49–56.
- [16] C. C.-Y. Lin and Y.-W. Chang, “ILP-based pin-count aware design methodology for microfluidic biochips,” in *Proc. IEEE/ACM DAC*, Jun. 2009, pp. 258–263.

- [17] R.-W. Liao and T.-H. Yang, "Design and fabrication of biochip for in-situ protein synthesis," M.S. thesis, Dept. Opt. Photonics, Natl. Central Univ., Jhongli, Taiwan, 2008.
- [18] M. G. Pollack, A. D. Shenderov, and R. B. Fair, "Electrowetting-based actuation of droplets for integrated microfluidics," *Lab Chip*, vol. 2, pp. 96–101, Mar. 2002.
- [19] F. Su and K. Chakrabarty, "Architectural-level synthesis of digital microfluidics-based biochips," in *Proc. IEEE/ACM ICCAD*, Nov. 2004, pp. 223–228.
- [20] F. Su, K. Chakrabarty, and R. B. Fair, "Microfluidics based biochips: Technology issues, implementation platforms, and design-automation challenges," *IEEE Trans. Comput.-Aided Design*, vol. 25, no. 2, pp. 211–223, Feb. 2006.
- [21] F. Su, W. Hwang, and K. Chakrabarty, "Droplet routing in the synthesis of digital microfluidic biochips," in *Proc. IEEE/ACM DATE*, Mar. 2006, pp. 1–6.
- [22] F. Su and K. Chakrabarty, "Module placement for fault-tolerant microfluidics-based biochips," *ACM TODAES*, vol. 11, no. 2, pp. 682–710, 2006.
- [23] V. Srinivasan, V. K. Pamula, and R. B. Fair, "An integrated digital microfluidic lab-on-a-chip for clinical diagnostics on human physiological fluids" *Lab Chip*, vol. 4, pp. 310–315, May 2004.
- [24] T. Xu and K. Chakrabarty, "Droplet-trace-based array partitioning and a pin assignment algorithm for the automated design of digital microfluidic biochips," in *Proc. IEEE/ACM CODES+ISSS*, Nov. 2006, pp. 112–117.
- [25] T. Xu and K. Chakrabarty, "Integrated droplet routing in the synthesis of microfluidic biochips," in *Proc. ACM/IEEE DAC*, Jun. 2007, pp. 948–953.
- [26] T. Xu and K. Chakrabarty, "Broadcast electrode-addressing for pin-constrained multi-functional digital microfluidic biochips," in *Proc. ACM/IEEE DAC*, Jun. 2008, pp. 173–178.
- [27] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang, "Placement of defect-tolerant digital microfluidic biochips using the T-tree formulation," *ACM Journal on Emerging Technol. Comput. Syst.*, vol. 3, no. 3, article 13, Nov. 2007.
- [28] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang, "BioRoute: A network flow based routing algorithm for digital microfluidic biochips," in *Proc. IEEE/ACM ICCAD*, Nov. 2007, pp. 752–757.
- [29] Y. Zhao, R. Sturmer, K. Chakrabarty, and V. K. Pamula, "Synchronization of concurrently-implemented fluidic operations in pin-constrained digital microfluidic biochips," in *Proc. IEEE VLSI Design*, Jan. 2010, pp. 69–74.
- [30] Y. Zhao and K. Chakrabarty, "Simultaneous optimization of droplet routing and control-pin mapping to electrodes in digital microfluidic biochips," *IEEE Trans. Comput.-Aided Design*, vol. 31, no. 2, pp. 242–254, Feb. 2012.



**Jia-Weng Chang** received the B.S. and M.S. degrees in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, in 2010 and 2012, respectively.

His current research interests include design automation on digital microfluidic biochips.



**Sheng-Han Yeh** received the B.S. degree in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, in 2011.

He is currently a Graduate Student with the Department of Computer Science and Information Engineering, National Cheng Kung University. His current research interests include physical design and design automation on digital microfluidic biochips.



**Tsung-Wei Huang** received the B.S. and M.S. degrees in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, in 2010 and 2011, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Texas, Austin.

His current research interests include design automation for reconfigurable architecture such as digital microfluidic biochips and field-programmable gate arrays.

Mr. Huang was a recipient of the Award of Microelectronics and Computer Development Fellowship. In 2011, he was a recipient of the First Prize from ACM SIGDA Student Research Competition (SRC), awarded at ACM/IEEE DAC. He was also the Second Prize Winner in ACM SRC Grand Final in 2011.



**Tsung-Yi Ho** (M'08) received the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2005.

Since 2007, he has been with the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan, where he is currently an Associate Professor. From 2003 to 2004, in 2005, and 2008, he was a Visiting Scholar with the University of California, Santa Barbara, Waseda University, Tokyo, Japan, Synopsys, Mountain View, CA, respectively. He has published

several papers in top journals and conferences, such as IEEE TCAD, ACM TODAES, ACM/IEEE DAC, IEEE/ACM ICCAD, ACM ISPD. His current research interests include design automation for digital microfluidic biochips and nanometer integrated circuits.

Dr. Ho was the recipient of many research awards, such as the Dr. Wu Tai-You Memorial Award of the National Science Council (NSC) of Taiwan (the most prestigious award from NSC for junior researchers), the Distinguished Young Scholar Award of the Taiwan IC Design Society, the ACM Taipei Chapter Young Researcher Award, the IEEE Tainan Chapter Gold Member Award, the Invitational Fellowship of the Japan Society for the Promotion of Science, Japan, and the Humboldt Research Fellowship from the Alexander von Humboldt Foundation, Germany.