Defect-Aware High-Level Synthesis and Module Placement for Microfluidic Biochips

Tao Xu, Krishnendu Chakrabarty, *Fellow, IEEE*, and Fei Su

***Abstract*—Recent advances in microfluidics technology have led to the emergence of miniaturized biochip devices, also referred to as lab-on-a-chip, for biochemical analysis. A promising category of microfluidic biochips relies on the principle of electrowetting-on-dielectric, whereby discrete droplets of nanoliter volumes can be manipulated using an array of electrodes. As chemists adapt more bioassays for concurrent execution on such "digital" droplet-based microfluidic platforms, system integration, design complexity, and the need for defect tolerance are expected to increase rapidly. Automated design tools for defect-tolerant and multifunctional biochips are important for the emerging marketplace, especially for low-cost, portable, and disposable devices for clinical diagnostics. We present a unified synthesis method that combines defect-tolerant architectural synthesis with defect-aware physical design. The proposed approach allows architectural-level design choices and defect-tolerant physical design decisions to be made simultaneously. We use a large-scale protein assay and the polymerase chain reaction procedure as case studies to evaluate the proposed synthesis method. We also carry out simulations based on defect injection to evaluate the robustness of the synthesized biochip designs.**

***Index Terms*—Lab-on-a-chip, microfluidics.**

I. Introduction

MICROFLUIDICS technology and bio-microelectromechanical systems (MEMS) have seen tremendous growth in the past few years [1]–[6]. A major application driver for microfluidics is biochemical analysis and fluidic processing in miniaturized devices, which are referred to in the literature as biochips or lab-on-a-chip. Biochips can be used for immunoassays, high-throughput sequencing, proteomic analysis involving proteins and peptides, and environmental toxicity monitoring. Another emerging application area for microfluidics-based biochips is clinical diagnostics, especially immediate point-of-care diagnosis of diseases [7].

A popular class of microfluidic biochips is based on continuous fluid flow in permanently etched microchannels. Fluid flow in these devices is controlled either using micropumps and microvalves, or by electrical methods based on electrokinetics and electroosmosis [2]. An alternative category of microfluidic biochips relies on the principle of electrowetting-on-dielectric. Discrete droplets of nanoliter volume can be manipulated using an array of electrodes. Following the analogy of digital electronics, this technology is referred to as "digital microfluidics" [1]. Because each droplet can be controlled independently, these systems also have dynamic reconfigurability, whereby groups of unit cells in a microfluidic array can be reconfigured to change their functionality during the concurrent execution of a set of bioassays.

As chemists adapt more bioassays for concurrent execution on a digital microfluidic platform, system integration and design complexity are expected to increase steadily [8], [9]. A digital microfluidic biochip platform has been developed for on-chip gene sequencing through synthesis [10], which targets execution of 10^6 fluidic operations. Other digital microfluidic biochips are being designed for protein crystallization which requires concurrent execution of hundreds of operations, with only microliter sample consumptions. The chip size is also increasing sharply. A recently demonstrated droplet-based biochip embeds more than 600 000 20 μm by 20 μm electrodes [11]. Moreover, next-generation biochips are expected to be multifunctional and adaptive "biochemical processing" devices. For example, inexpensive biochips for clinical diagnostics integrate hematology, pathology, molecular diagnostics, cytology, microbiology, and serology onto the same platform. The emergence of such integrated, multifunctional, and reconfigurable platforms provides the electronic design automation community with a new application driver and market for research into new algorithms and design tools.

As in the case of integrated circuits (ICs), increase in density and area of microfluidics-based biochips will reduce yield, especially for new technology nodes. Low yield will deter large-scale and high-volume production, and it will increase production cost. It will take time to ramp up yield learning based on an understanding of defect types in such mixed-technology devices. Therefore, defect-tolerant designs are important for the emerging marketplace, especially for low-cost, portable, and disposable devices for clinical diagnostics.

Another reason for the importance of defect tolerance lies in the projected use of microfluidic biochips for safety-critical applications, e.g., patient health monitoring, neo-natal care, and the monitoring of environmental toxins. Therefore, defect tolerance must be integrated into the automated design tools to ensure high levels of system dependability. Despite the recent emergence of automated synthesis methods for biochips [9], [12], [13], defect tolerance has largely been overlooked in the literature.

Manuscript received January 4, 2008; revised January 28, 2008. This work was supported in part by the National Science Foundation under Grant IIS-0312352 and Grant CCF-0541055. A preliminary version of this paper was presented at the 2007 IEEE International Conference on VLSI Design. This paper was recommended by S. DeWeerth.

T. Xu and K. Chakrabarty are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 22708 USA (e-mail: tx@ee.duke.edu; krish@ee.duke.edu).

F. Su is with the Intel Corporation, Folsom, CA 95630 USA (e-mail: fei.su@intel.com).

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/TBCAS.2008.918283

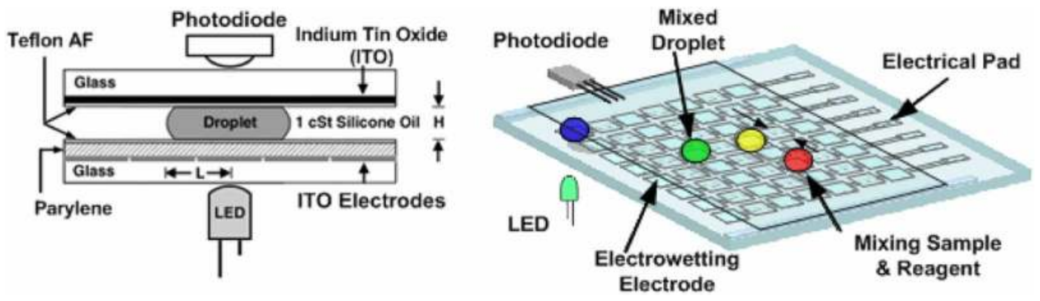


Fig. 1. Schematic of a digital microfluidic biochip [14].

As in traditional IC design, the automated design of a digital microfluidic biochip can be divided into two major phases. High-level synthesis is first used to generate a macroscopic structure of the biochip from the behavioral model for a bioassay; this structure is analogous to a structural register-transfer level model in electronics computer-aided design (CAD). Physical design creates the final layout of the biochip, consisting of the placement of microfluidic modules such as mixers and storage units, as well as the routes that droplets take between different modules, locations of on-chip reservoirs, dispensing ports and integrated optical detectors, and other geometrical details. These synthesis tools can relieve biochip users from the burden of manually optimizing a set of assays for increased throughout. Users can describe bioassays at a sufficiently high level of abstraction; synthesis tools can then map the behavioral description to the microfluidic array and generate an optimized schedule of bioassay operations, the binding of assay operations to resources, and a layout of the microfluidic biochip. Thus, the biochip user can concentrate on the development of the nano- and micro-scale bioassays, leaving implementation details to the synthesis tools.

In the past few years, several automated design tools have been proposed for microfluidic biochips. These design automation methods address operation scheduling and module placement for digital microfluidics. However, most of these tools have been focused on device-level physical modeling and simulation of single components [3]. Few efforts have been reported on developing system-level CAD tools for digital microfluidic biochips design. Moreover, defect tolerance has largely been overlooked in these tools.

In this paper, we propose a synthesis methodology that unifies operation scheduling, resource binding, module placement, and defect tolerance. The proposed automated design technique is based on parallel recombinative simulated annealing (PRSA) and its uses defect tolerance as a design criterion. All three tasks, i.e., resource binding, scheduling, and placement, are carried out at each step of the algorithm in a defect-aware fashion. Thus, exact placement information, instead of a crude area estimate, is used to judge the quality of architectural-level synthesis. This information is utilized by the annealing process to select resources and schedule bioassay operations to produce a high-quality design. The proposed method allows defect-tolerant architectural design choices and defect-aware physical design decisions to be made simultaneously. Defect tolerance is incorporated during synthesis since it is integrated into the simulated annealing procedure. We use a representative protein assay the polymerase chain reaction (PCR) procedure to evaluate the proposed synthesis methodology.

The rest of the paper is organized as follows. Section II provides an overview of digital microfluidic biochips. In Section III we discuss related prior work on biochip design automation. Section IV presents an overview of a defect-oblivious unified synthesis method. In Section V, we introduce a new criterion of evaluating defect tolerance for a digital microfluidic biochip and incorporate it in an enhanced version of the synthesis technique. In Section VI, we use a protein assay and PCR as case studies to evaluate the proposed synthesis method. We also carry out simulations based on defect injection to evaluate the robustness of the synthesized biochip designs. Finally, conclusions are drawn in Section VII.

II. Digital Microfluidic Biochips

The microfluidic biochips discussed in this paper are based on the manipulation of nanoliter droplets using the principle of electrowetting, i.e., modulation of the interfacial tension between a conductive fluid and a solid electrode through an electric field. The basic cell of a digital microfluidic biochip consists of two parallel glass plates, as shown in Fig. 1. The bottom plate contains a patterned array of individually controllable electrodes, and the top plate is coated with a continuous ground electrode. The droplets containing biochemical samples and the filler medium, such as the silicone oil, are sandwiched between the plates. The droplets travel inside the filler medium. By varying the electrical potential along a linear array of electrodes, droplets can be moved along this line of electrodes. The velocity of the droplet can be controlled by adjusting the control voltage (0–90 V), and droplets can be moved at speeds of up to 20 cm/s [1]. Based on this principle, microfluidic droplets can be moved freely to any location of a 2-D array without the need for micropumps and microvalves.

Using a 2-D array, many basic microfluidic operations for different bioassays can be performed, such as sample introduction (*dispense*), sample movement (*transport*), temporarily sample preservation (*store*), sample dilution with buffer (*dilute*) and the mixing of different samples (*mix*). For instance, the *mix* operation is used to route two droplets to the same location and then turn them around some pivot points. Note that these operations can be performed anywhere on the array, whereas in continuous-flow biochips they must operate in a specific micromixer or microchamber fixed on a substrate. This property is referred to as the reconfigurability of a digital microfluidic biochip. The configurations of the microfluidic array, i.e., the routes that droplets transport and the rendezvous points of droplets, are programmed into a microcontroller that controls the voltages of electrodes in the array. In this sense, these mixers and storage units used during the operations can be viewed as reconfigurable virtual devices. In addition, a digital microfluidic biochip also contains on-chip reservoirs/dispensing ports that are used to generate and dispense sample or reagent droplets, as well as integrated optical detectors such as LEDs and photodiodes. In contrast to mixers or storage units, these resources are nonreconfigurable.

III. Related Prior Work

Biochips belong to the class of MEMS, which is a relatively young field compared to IC design. Nevertheless, a number of MEMS CAD tools are available today for modeling [15] and synthesis [16]. Attempts have also been made to make MEMS defect-tolerant [17]. However, because of the differences in actuation methods between MEMS and digital microfluidics, these techniques cannot be directly used for the design of microfluidic biochips.

Although research on microfluidic biochips has gained momentum in recent years, CAD tools for biochips are still in their infancy [18]. Recent years have seen growing interest in the this area [9], [12], [13], [19], [20]. One of the first published methods for biochip synthesis decouples high-level synthesis from physical design [13]. It is based on rough estimates for placement costs such as the areas of the microfluidic modules. These estimates provide lower bounds on the exact biochip area, since the overheads due to spare cells and cells used for droplet transportation are not known *a priori*. However, it cannot be accurately predicted if the biochip design meets system specifications, e.g., maximum allowable array area and upper limits on assay completion times, until both high-level synthesis and physical design are carried out. When design specifications are not met, time-consuming iterations between high-level synthesis and physical design are required. A link between these steps is especially necessary if defect tolerance is to be considered during synthesis. Moreover, defect-tolerance must be incorporated to guarantee the reliability of the synthesis result.

IV. Unified Synthesis Method

A. Problem Formulation

Fig. 2 illustrates the design flow for the proposed synthesis method. A sequencing graph is first obtained from the protocol for a bioassay [4]. This acyclic graph $G(V,E)$ has vertex set $V = \{v_i : i = 0,1,\ldots,k\}$ in one-to-one correspondence with the set of assay operations, and edge set $E = \{(v_i,v_j) : i,j = 0,1,\ldots,k\}$ representing dependencies between assay operations. The weight for each node, $d(v_i)$, denotes the time taken for operation v_i; note however that this value is not assigned until resource binding has been performed during synthesis. Since droplet movement is very fast in contrast to assay operations [1], [21], we can ignore the droplet transportation time between different assay operations. In addition, a microfluidic module library is also provided as an input of the synthesis procedure. This 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. Each module is characterized by its function (mixing, storing, detection, etc.) and parameters such as width, length and operation duration. Moreover, some design specifications are also given *a priori*, e.g., an upper limit on the bioassay completion time $T_{\max}$, an upper limit on the size of microfluidic array $A_{\max}$, and the set of nonreconfigurable resources such as on-chip reservoirs/dispensing ports and integrated optical detectors.

The proposed synthesis tool performs scheduling, resource binding, and placement in a unified manner. As in the case of high-level synthesis for ICs, resource binding refers to the mapping from bioassay operations to available functional resources. Note that there may be several types of resources for any given bioassay operation. For example, a 2×2-array mixer, a 2×3-array mixer and a 2×4-array mixer can be used for a droplet mixing operation [1]. These mixers differ in their areas as well as mixing times. In such cases, a resource selection procedure must be used. On the other hand, due to the resource constraints, a resource binding may associate one functional resource with several assay operations; this necessitates resource sharing. Once resource binding is carried out, the time duration for each bioassay operation can be easily determined. Scheduling determines the start times and stop times of all assay operations, subject to the precedence constraints imposed by the sequencing graph. In a valid schedule, assay operations that share a microfluidic module cannot execute concurrently. Scheduling and resource binding also need to be tied to the placement problem for biochips; placement determines the various configurations of a microfluidic array as well as the locations of integrated optical detectors and reservoirs/dispensing ports. The property of virtual devices makes the placement of reconfigurable microfluidic modules, such as mixers or storage units, on a 2-D microfluidic array quite different from the traditional placement problem in VLSI design.

The output of the synthesis tool includes the mapping of assay operation to resources, a schedule for the assay operations, and the placement of the modules. The synthesis procedure attempts to find a desirable design point that satisfies the input specifications. If such a solution does not exist, the synthesis tool outputs the best solution that can be achieved. In order to measure the quality of a synthesis flow, we need to consider the minimization of the array area A and the completion time T for the bioassay. For this multi-objective optimization problem, a weighting approach is used. Here weights α and $(1-\alpha)$, where $0 < \alpha < 1$, are assigned to the criteria of normalized area (denoted by $A/A_{\max}$) and normalized bioassay time (denoted by $T/T_{\max}$), respectively. The solution with the lowest value of the metric $(\alpha \times A/A_{\max} + (1-\alpha) \times T/T_{\max})$ is considered to be an acceptable solution.

B. PRSA-Based Algorithm

The resource-constrained scheduling problem and the module placement problem have been shown in the literature

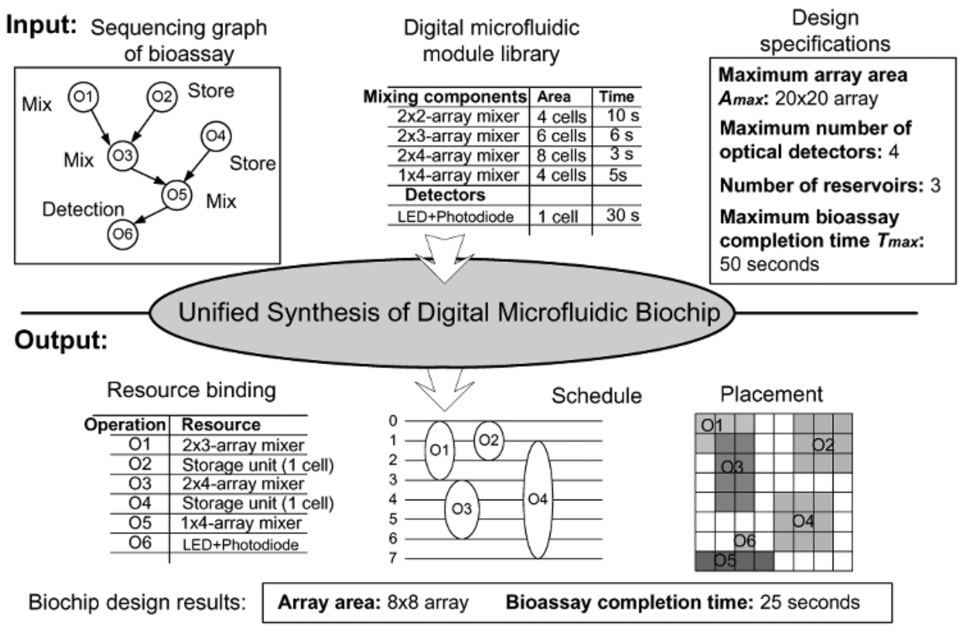

Input: Sequencing graph of bioassay
Mix O1
O2 Store
Mix O3
O4 Store
O5 Mix
Detection O6
Digital microfluidic module library
Mixing components | Area | Time
2x2-array mixer | 4 cells | 10 s
2x3-array mixer | 6 cells | 6 s
2x4-array mixer | 8 cells | 3 s
1x4-array mixer | 4 cells | 5s
Detectors
LED+Photodiode | 1 cell | 30 s
Design specifications
Maximum array area A_{max}: 20x20 array
Maximum number of optical detectors: 4
Number of reservoirs: 3
Maximum bioassay completion time T_{max}: 50 seconds
Unified Synthesis of Digital Microfluidic Biochip
Output:
Resource binding
Operation | Resource
O1 | 2x3-array mixer
O2 | Storage unit (1 cell)
O3 | 2x4-array mixer
O4 | Storage unit (1 cell)
O5 | 1x4-array mixer
O6 | LED+Photodiode
Schedule
Placement
Biochip design results: Array area: 8x8 array Bioassay completion time: 25 seconds

Fig. 2. Example illustrating system-level synthesis [13].

to be *NP*-complete [22]. Therefore, heuristics are needed to solve the optimization problem in a computationally efficient manner. In this paper, we propose a synthesis algorithm based on simulated annealing, which is widely used in traditional electronic design automation [23]. Since we are dealing with multi-objective optimization (chip area, assay time, defect tolerance), we combine simulated annealing algorithm with a genetic algorithm to better represent candidate designs, as has been done earlier in electronic design automation [24]. We therefore focus on a parallel recombinative simulated annealing (PRSA) based algorithm. PRSA is a well-studied combinational optimization method that has some of the best attributes of both genetic algorithms and simulated annealing algorithms [25]. This class of algorithms is best viewed as genetic algorithms that use *Boltzmann trials* between modified and existing solutions to select the solutions that exist in the next generation.

We present a PRSA-based algorithm to solve the optimization problem for biochip synthesis. The pseudocode for this heuristic approach is shown in Fig. 3. Some details of the procedure are listed as follows, and they also will be illustrated in Section V.

1) Representation of a Chromosome: A robust representation technique called *random keys* is used in this algorithm [26]. A random key is a random number sampled from $[0, 1]$. Each chromosome in the population can be encoded as a vector of random keys, named genes.$Chromosome = \{\text{gene}(1), \ldots, \text{gene}(k), \text{gene}(k+1), \ldots, \text{gene}(2k), \text{gene}(2k+1), \ldots, \text{gene}(3k)\}$, where k is the number of assay operations. Here the first set of k genes are used to determine resource binding, i.e., $Rb(i) = \text{gene}(i)$, $i = 1$ to k. The second set of k genes are to set the delay time of the operations, which is calculated as follows: delay value of operation $Dv(i) = d \times \text{gene}(i+k)$, $i = 1$ to k, where d is a constant that can be fine-tuned through experiments. The last k genes are used to determine the placement priorities, i.e., priority value of operation $Pv(i) = \text{gene}(i+2k)$, $i = 1$ to k.

```
PRSA-based algorithm
1  Set initial population of chromosome and the initial temperature $T_\infty$;
2  Implement the synthesis using the information of initial chromosomes:
     {Phase I: Resource binding; Phase II: Scheduling; Phase III: Placement}
3  while (Stopping criteria of annealing is not satisfied)
4    for $i = 1: N$ /* Inner loop of annealing process */
5      Find fitness values of chromosomes through construction procedure;
6      Reproduction; /* Best chromosomes copied to the next generation */
7      Crossover: {Parameterized uniform crossover is to generate the child
             chromosome from two randomly-selected parent chromosomes}
8          if Fitness(child) < Fitness(parents) or
                 rand(0, 1) < exp(-[Fitness(child)–Fitness(parents)]/T)
9             Child chromosome is selected;
10         else Parent chromosome (the best one) is selected;
11         end if  /* Here a Boltzmann trial is performed */
12     Mutation; /* New chromosomes are generated randomly */
13     New population replaces the old generation; end for
14   T= rate × T; /* update the temperature */ end while
15 Find the best chromosome from the final population;
16 Output the results of resource binding, scheduling and placement.
```

Fig. 3. Pseudo-code for the PRSA-based heuristic algorithm.

2) Construction Procedure: The goal of this procedure is to carry out resource binding, scheduling and placement under dependency and resource constraints, by using a vector of random numbers (i.e., genes from a chromosome). It consists of the following three phases.

1) Phase I—*Resource binding*: To simplify the synthesis procedure, in this phase we temporarily do not consider an upper limit on the number of available reconfigurable resources. A reconfigurable resource type for a bioassay is selected based on its associated gene value, i.e., $Rb(i)$. For example, for a mixing operation v_i, a 2×2-array mixer is selected if $Rb(i) < 0.25$; a 2×3-array mixer is chosen

if $0.25 \leq Rb(i) < 0.5$; a 2×4-array mixer is selected if $0.5 \leq Rb(i) < 0.75$; a 4-electrode linear array mixer is selected if $Rb(i) \geq 0.75$.
Reservoirs/dispensing ports and optical detectors are nonreconfigurable resources. The number of such resources is fixed, and it is determined by the system design specifications. The gene values for the corresponding operations determine the selection of resource instance. For example, if there are two optical detectors available, namely OD_1 and OD_2, a optical detection operation v_i is bound to OD_1 if $Rb(i) < 0.5$, and to OD_2 if $Rb(i) \geq 0.5$.
After Phase I, a weight $d(v_i)$, i.e., the duration time for the corresponding operation, has been assigned to each node v_i of the sequencing graph. Thus, an original sequencing graph without node weights is modified to a weighted sequencing graph.

2) Phase II—*Scheduling*: In this phase, a feasible bioassay schedule, satisfying temporal precedence constraints as well as nonreconfigurable resource constraints, is constructed by using the delay values $Dv(i)$ from a chromosome. Due to its low computational complexity of $O(n)$, where n is the number of operations to schedule, a *list scheduling* algorithm is used in this step [27]. As in Phase I, only constraints for nonreconfigurable resources are taken into account here.
To schedule the operation i, we set its start time to be $\mathrm{Start}(i) = \max\{\mathrm{Stop}(j) : \text{either } j \text{ is the predecessor of } i, \text{ or it used the same resource as } i\} + Dv(i)$ and stop time as $\mathrm{Stop}(i) = \mathrm{Start}(i) + d(v_i)$. After this phase, a scheduled sequencing graph with resource binding is obtained.

3) Phase III—*Placement*: Based on the results from resource binding and scheduling, we attempt to place the microfluidic modules on a 2-D array to satisfy the design specifications. A greedy algorithm referred to as *Kamer-BF* algorithm is used in this phase [28]. Microfluidic modules are first sorted in the descending order of their priority values $Pv(i)$. In each step, the module with the highest priority among the unplaced ones is selected and placed. To minimize the chip area, the selected module can only be placed adjacent to modules which have already been located on the chip layout to avoid waste of space. Note that there might be multiple locations for the selected module. In this case, the greedy algorithm evaluates each placement and selects the one which result in the smallest chip area. Resource constraints must be satisfied, e.g., there should be no spatial overlap between the module with previously placed ones if their usage overlaps in the schedule. The placement problem can also be modeled by a 3-D packing problem, which will be illustrated by an example in Section V. In addition, we add a segregation region between two active modules. This additional area not only isolates the functional module from its neighbors, thereby avoiding unexpected cross-contamination, but it also provides a transportation path for droplet movement between different modules.

The above greedy algorithm not only deals with the placement of reconfigurable resources, but it can also adapt to the location of nonreconfigurable resources such as optical detectors. For a fabricated chip, the locations of the optical detectors are fixed. The placement algorithm views these optical detectors as preplaced modules and ensures that the location of optical detectors will not be used by other modules during their scheduled operation time. On the other hand, the locations of reservoirs/dispensing ports can be determined manually after synthesis, since they do not affect the area of microfluidic array or the processing time for the bioassay.

Therefore, based on the information provided by a chromosome, the synthesis procedure can be carried out based on the above three phases. The fitness value of this chromosome is determined by the synthesis results. Through a series of generations of evolution controlled by a simulated annealing process, we can find a best chromosome, i.e., with the smallest fitness value, from the final population. The synthesis results obtained from this chromosome represent the solution to our optimization problem.

V. Defect-Tolerant Synthesis

Digital microfluidic biochips are fabricated using standard microfabrication techniques [1]. Due to the underlying mixed technology and multiple energy domains, they exhibit unique failure mechanisms and defects. A manufactured microfluidic array may contain several defective cells. We have observed defects such as dielectric breakdown, shorts between adjacent electrodes, and electrode degradation, particle contamination and residue, etch variations, etc. [29], [30].

Reconfiguration techniques can be used to bypass faulty cells or faulty optical detectors to tolerate manufacturing defects. Bioassay operations bound to these faulty resources in the original design need to be remapped to other fault-free resources. Due to the strict resource constraints in the fabricated biochip, alterations in the resource binding operation, schedule and placement must be carried out carefully. Our proposed system-level synthesis tool can be modified to deal with this issue by introducing defect tolerance schemes.

To reconfigure a defective biochip, a PRSA-based algorithm along the lines of the one described in Section IV-B was used in [31]. The following additional considerations must be taken into account.

1) The objective during reconfiguration is to minimize the bioassay completion time while accommodating all microfluidic modules and optical detectors in the fabricated microfluidic array.
2) As resource constraints, the defect-free parts of the microfluidic array and the number of fabricated fault-free nonreconfigurable resources replace the original design specifications.
3) In the placement phase, the locations of the defective cells are no longer available. Note that the locations of nonreconfigurable resources such as integrated optical detectors and reservoirs/ dispensing ports are fixed in the fabricated biochip.

Using this enhanced synthesis tool, a set of bioassays can be easily mapped to a biochip with a few defective cells; thus we do not need to discard the defective biochip.

The enhanced defect tolerance schemes in this paper are composed of two attributes: defect-aware synthesis, i.e., anticipate

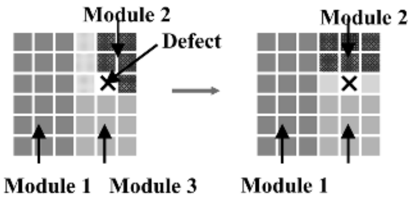


Fig. 4. Example of partial reconfiguration.

defect occurrences and design the system to be defect-resilient, and postmanufacture reconfiguration and re-synthesis. In this section, we incorporate defect tolerance in the unified synthesis flow for microfluidic biochips. A novel partial reconfiguration method is also presented to enhance defect tolerance after the device is manufactured.

A. Defect-Aware Synthesis

We first focus on anticipatory defect-aware synthesis, whereby we provide guarantees on correct bioassay operation even if the manufacturing process introduces defects. Instead of handling defects after they are detected, we attempt to achieve defect-tolerant mapping of bioassay protocols to the microfluidic array under broad assumptions of defect occurrences.

1) Defect Tolerance Index: Defect tolerance of a synthesized biochip can be evaluated in terms of survivability, i.e., the capability to perform bioassays on a microfluidic array with defects. The defect tolerance index (DTI), is defined as the probability that defect tolerance can be achieved via successful partial reconfiguration when the array contains defective cells [29]. Partial reconfiguration refers to the relocation only of the modules that contain defective cells; other modules are not affected. The relocated modules therefore "survive" through the defects (see Fig. 4).

Assume that each cell in the microfluidic array has an independent failure probability p. The DTI $D(G)$ value for a layout G can be estimated by multiplying the survival probabilities of all the modules, as follows [31]:

$$D(G) \approx \prod Ps(M_i) = \prod (1 - f_1(M_i) + f_1(M_i) \times f_2(M_i))$$

where M_i, $i = 1 \ldots N$, is a microfluidic module (e.g., mixer) contained in a given layout G, and $Ps(M_i)$ is the survival probability of module M. Note that $f_1(M_i)$ is the probability that the module M_i is faulty. It is determined by the equation $f_1(M_i) = 1 - q^{A(Mi)}$, where $q = 1 - p$ and $A(M_i)$ is the total number of cells contained in M_i. Finally, $f_2(M_i)$ is the probability that M_i can be successfully reconfigured if it becomes faulty [24].

It is not trivial to accurately determine the value of $f_2(M_i)$. Instead of invoking complicated procedures involving fault simulation, we simply examine the biochip configuration (e.g., its empty spaces) and estimate the ease of reconfiguration. For example, partial reconfiguration for module M_i is easier if the maximum empty rectangle (MER) for M_i is relatively large. Thus, $f_2(M_i)$ increases with the ratio of MER size to the area of M_i, that is, with $R(M_i) = A(\mathrm{MER})/A(M_i)$. Thus, we can estimate the value of $f_2(M_i)$ using a simple function of $R(M_i)$ and other variables. For example, $f_2(M_i) \approx (1-q^{A(Mi)})^{R(Mi)}$, whereby we divide the MER for M_i into $R(M_i)$ clusters, and then the reconfiguration probability $f_2(M_i)$ for M_i can be determined by the likelihood of having at least one fault-free empty cluster. We can further include some constants into the previous function: $f_2(M_i) \approx 1 - k_1(1 - (k_2 q)^{k3A(Mi)})^{R(Mi)}$, where constants $k_1 \sim k_3$ can be fine tuned through experiments [29].

Now we incorporate DTI into the PRSA-based unified synthesis method. We first define layout vulnerability by $V = 1 - D$. Layouts with low vulnerability are likely to provide high probability of successful partial reconfiguration. To find such designs, we combine vulnerability with time- and area-cost to derive a new fitness function to control the PRSA-based procedure. Candidate designs with low survivability are discarded during evolution. Thus, the synthesis procedure anticipates defect occurrences and selects designs that allow reconfiguration of large number of modules, while meeting constraints on array size and bioassay processing time.

B. Partial Reconfiguration and Partial Resynthesis

Next we discuss how defects can be bypassed after manufacture. The defect-aware synthesis method described in Section V-A attempts to ensure the availability of unused cells in the microfluidic array to avoid defective cells that are located after manufacture. Here we propose an efficient method to achieve defect tolerance via partial reconfiguration (introduced in Section V-A) using these unused defect-free cells. For each affected module, we search the array for available defect free areas for partial reconfiguration. This can be accomplished fast, because the search space is restricted to the layouts in the modules' time duration. For each layout, we use a staircase algorithm from the literature [31] to search for the maximal-empty rectangles (MER) in the microfluidic array, and then check if these rectangles can accommodate the faulty module [31].

Once a module is relocated, the algorithm updates the corresponding layout and starts the search for the next module. Resources binding and scheduling results are not changed. Only the placement of defective modules is modified. Moreover, if the number of defective cells is not excessive, most microfluidic modules on the array are not affected and they do not need not to be reconfigured. As discussed in the Section V-A, the incorporation of defect tolerance in the design flow ensures a high probability of partial reconfigurability of the modules, i.e., it is very likely that the defective biochip can be made usable via partial reconfiguration, which can be accomplished very fast.

However, such a partial reconfiguration procedure might not be feasible because of lack of availability of spare cells. In some cases, there may be not enough defect free cells to carryout partial reconfiguration for some defective modules. In such scenarios, the fabricated biochip must be discarded. A straight forward solution to this problem is as follows. After defects are identified, the complete synthesis process is repeated to generate a new design using only defect-free cells. However, this approach imposes additional computation burden on the design and implementation process.

Defect tolerance is achieved by complete resynthesis, which can be very time consuming. We therefore introduce a new method called partial resynthesis. The key idea here is to truncate the bioassay and carry out resynthesis only for the modules

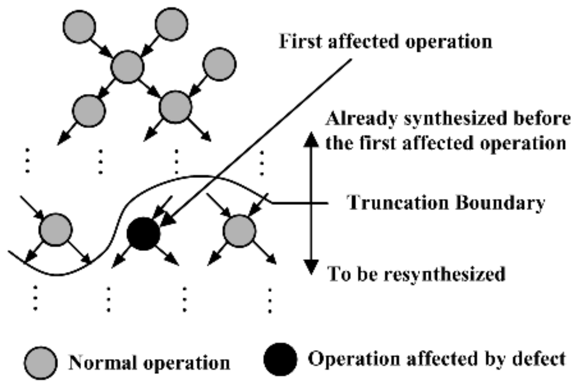

First affected operation
Already synthesized before the first affected operation
Truncation Boundary
To be resynthesized
Normal operation
Operation affected by defect

Fig. 5. Illustration of partial resynthesis.

that start later than the earliest-in-use defective module, see Fig. 5. To carry out truncation, we first find the earliest-in-use defective module. Then we look at the placement layout at the time when the affected module is scheduled to start. All modules placed at that time instance form the truncation boundary. The sequencing graph is then be truncated from that boundary. Synthesis results for the segment of bioassay before the truncation boundary are not affected by defects, thereby no change is needed. Only the segment after the truncation boundary is resynthesized. To further reduce the computational complexity, the bioassay can be truncated only from the top but also from the bottom of the sequencing graph, i.e., from the earliest-in-use defective module to the latest. By this means, the synthesis tool precisely identifies the defective parts of the design, resynthesizes it, and recombines the synthesis results. The start and end point for each synthesized design component must be carefully adjusted in the recombination procedure. Also, since resynthesis is carried out on a defective chip, which leaves a smaller chip area, it may generate a new design for the truncated part, with a longer completion time than for the previous design. Therefore, additional time slack may be added if necessary. Recall that in the scheduling phase of the PRSA-based method, the starting time of a module is obtained by adding a delay to the time that its predecessor is completed. This delay is a randomly generated value with an upper bound of $D_{\max}$. Therefore, we can introduce the time slack by increasing $D_{\max}$.

Although the above partial resynthesis procedure may take as much time as complete resynthesis in the worst case, i.e., if both the first and last in-use module is defective and cannot be relocated, it is faster on average than the complete-resynthesis procedure.

Using these two methods, the complexity of doing postmanufacture processing for defect tolerance can be greatly reduced compared to resynthesis. The time-cost for a set of bioassays is also significantly decreased.

VI. Experimental Evaluation

A. Synthesis Results

We evaluate the proposed defect-aware synthesis method by using it to design a biochip for a representative protein assay and the polymerase chain reaction (PCR) procedure.

Recently, the feasibility of performing a colorimetric protein assay on a digital microfluidic biochip has been successfully demonstrated [21]. Based on the Bradford reaction [21], the protocol for a generic droplet-based colorimetric protein assay is as follows. First, a droplet of the sample, such as serum or some other physiological fluid containing protein, is generated and dispensed into the biochip. Buffer droplets, such as 1M NaOH solution, are then introduced to dilute the sample to obtain a desired dilution factor (DF). This on-chip dilution is performed using multiple hierarchies of binary mixing/splitting phases, referred to as the interpolating serial dilution method [1]. The mixing of a sample droplet of protein concentration C and a unit buffer droplet results in a droplet with twice the unit volume, and concentration $C/2$. Splitting this large droplet results in two unit-volume droplets of concentration $C/2$ each. Continuing this step in a recursive manner using diluted droplets as samples, an exponential dilution factor of $\mathrm{DF} = 2^N$ can be obtained in N steps. After dilution, droplets of reagents, such as Coomassie brilliant blue G-250 dye, are dispensed into the chip, and they mix with the diluted sample droplets. Next the mixed droplet is transported to a transparent electrode, where an optical detector (e.g., a LED-photodiode setup) is integrated. The protein concentration can be measured from the absorbance of the products of this colorimetric reaction using a rate kinetic method [21]. Finally, after the assay is completed, all droplets are transported from the array to the waste reservoir.

A sequencing graph model can be developed from the above protocol for a protein assay $(\mathrm{DF} = 128)$, as shown in Fig. 6. There are a total of 103 nodes in one-to-one correspondence with the set of operations in a protein assay, where $DsS, DsB_i(i = 1,\ldots,39)$, and $DsR_i(i = 1,\ldots,8)$ represents the generation and dispensing of sample, buffer and reagent droplets, respectively. In addition, $Dlt_i(i = 1,\ldots,39)$ denotes the binary dilution (including mixing/splitting) operations, $\mathrm{Mix}_i(i = 1,\ldots,8)$ represents the mixing of diluted sample droplets, and reagent droplets; $\mathrm{Opt}_i(i = 1,\ldots,8)$ denotes the optical detection of the mixed droplets. Until the fourth step of a serial dilution, all diluted sample droplets are retained in the microfluidic array. After that stage, for each binary dilution step, only one diluted sample droplet is retained after splitting, while the other droplet is moved to the waste reservoir.

The basic operations for protein assay have been implemented on a digital microfluidic biochip [1], [21]. Experiments indicate that the dispensing operation takes 7 s when we use a reservoir of 4 mm diameter and a dispensing channel comprising 750-μm pitch electrodes with 100-μm channel gap [1]. Mixing is an important, yet difficult, microfluidic operation. Linear array mixing and 2-D array mixing have been performed on a biochip, and the operation times of various mixers have been found to be different [1]. Note that in these experiments, cells in mixers were assumed to have the same geometric parameters, i.e., a 1.5-mm electrode size and the 600-μm gap between the two plates. A binary dilution operation can also be easily implemented by using a linear array or 2-D array, whereby the mixing of sample droplet and buffer droplet is followed by droplet splitting. Absorbance of the assay product can be measured using an integrated LED-photodiode setup.

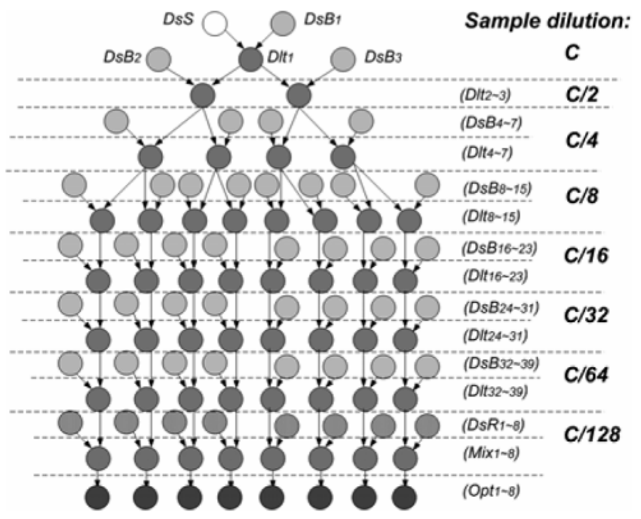


Fig. 6. Sequencing graph for a protein assay.

TABLE I
MODULE LIBRARY FOR SYNTHESIS

Operation	Resource	Time (s)
DsS; *DsB*; *DsR*	On-chip reservoir/dispensing port	7
Dlt	2x2-array dilutor	12
	2x3-array dilutor	8
	2x4-array dilutor	5
	4-electrode linear array dilutor	7
Mix	2x2-array mixer	10
	2x3-array mixer	6
	2x4-array mixer	3
	4-electrode linear array mixer	5
Opt	LED+Photodiode	30
Storage	Single cell	N/A

Experiments indicate this absorbance measurement takes 30 s [21]. Thus, we can build a microfluidic module library for a protein assay, as shown in Table I.

We also need to specify some design parameters for the biochip to be synthesized. As an example, we set the maximum microfluidic array size to be 10×10 cells, and the maximum allowable completion time for the protein assay to be 400 s. We assume that there is only one on-chip reservoirs/dispensing port available for sample fluids, but two such ports for buffer fluids, two for reagent fluids, and one for waste fluids. Finally, we assume that at most four optical detectors can be integrated into this biochip.

A second bioassay, namely the mixing stages of the PCR is also used in this paper. PCR is one of the most commonly used procedure for DNA analysis. It is used for rapid enzymatic amplification of specific DNA exponentially using temperature cycles. Recently, the feasibility of performing droplet-based PCR on digital microfluidics-based biochips has been successfully demonstrated [32]. Here we use the mixing stage of PCR as an example to evaluate the defect tolerance capability of defect-oblivious and defect-aware synthesis methods. Its assay protocol can be modeled by a sequencing graph, as shown in Fig. 8. The resource library is shown in Table II. The data for the operation times associated with the different modules are obtained from real-life experiments. Unlike the protein assay,

TABLE II
MIXER LIBRARY FOR PCR MIXING STAGE

Hardware	Module	Mixing time
2x2 electrode array	4x4 cells	10s
2x3 electrode array	4x5 cells	6s
2x4 electrode array	4x6 cells	3s
4-electrode linear array	3x6 cells	5s

no design constraints (area-cost, time-cost) are provided for the PCR assay.

Before applying the proposed synthesis method to the above problem instance, we use two baseline techniques to design the biochips for protein assays. In the first design, we attempt to minimize the microfluidic array size as much as possible, as shown in Fig. 7(a). Only one linear array, i.e., a 4-electrode linear array, is used as both the mixer and the dilutor. It also provides the location for optical detection. Moreover, three additional storage units are needed to store the intermediate droplets, i.e., diluted samples. Due to the high resource constraint in this design, the operations of dilution, mixing and optical detection have to be carried out sequentially. Consequently, the completion time for the protein assay is as high as 560 s, which exceeds the design specification of 400 s. As a second baseline case, we attempt to minimize the assay processing time using the genetic algorithm that was proposed in [14]. In this method, only area estimates of the microfluidic array, i.e., the sum of the areas of active microfluidic modules in each time step, are used to guide the scheduling procedure. To minimize the operation time, 2×4-array modules are used for dilution and mixing. A completion time of 297 s for the protein assay is obtained using this method. However, due to the absence of exact placement information, this design cannot guarantee that spatial constraints on the design are satisfied. For example, it can be shown that at time step 167 s in this schedule, five 2×4-array dilutors as well as 14 storage units are active on the array simultaneously. Although their area estimate satisfies the resource constraint (i.e., $5 \times 8 + 14 = 54 < 10 \times 10$), we cannot pack these microfluidic modules without overlaps in a 10×10 array if we incorporate the segregation regions between modules, as shown in Fig. 7(b). This implies that the resulting design fails to meet the design specification related to array area.

We now use the PRSA-based algorithm described in Section IV-B to find a desirable solution that satisfies design specifications. For evaluation of defect tolerance, we synthesize the assay using the PRSA-based algorithm twice, i.e., with and without defect-tolerance incorporated.

First, the defect-oblivious version is used. In the simulation experiments, we set the number of chromosomes in the population to 103. During evolution, the ten best chromosomes are reproduced into the next generation. A total of 36 chromosomes in the new population result from the crossover. The remaining 57 chromosomes are obtained from the mutation operators, where 19 new chromosomes are from the mutation of genes involved with resource binding, 19 from the mutation of genes for scheduling, and 19 from the mutation of genes for placement. Here mutation is implemented by randomly generating the new random keys to replace the old ones. For the annealing scheme, the initial temperature is chosen to ensure that almost every

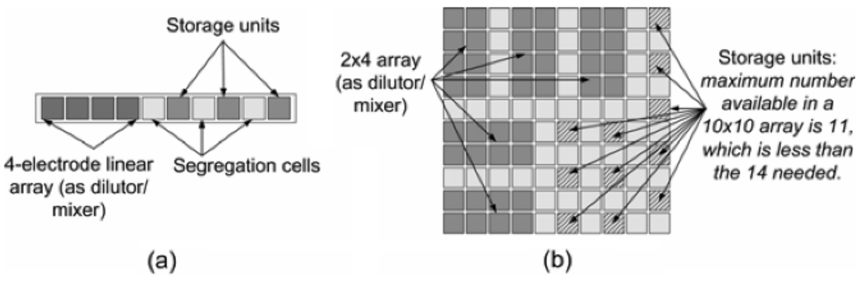

Storage units
4-electrode linear array (as dilutor/ mixer)
Segregation cells
(a)
2x4 array (as dilutor/ mixer)
Storage units: maximum number available in a 10x10 array is 11, which is less than the 14 needed.
(b)

Fig. 7. (a) Baseline case 1: a full-custom design. (b) Baseline case 2: violation of given specifications for the design obtained from [4].

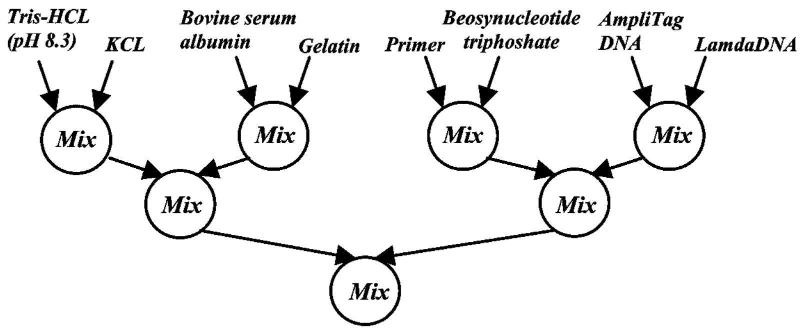


Fig. 8. Sequencing graph for the mixing stage of PCR.

new child chromosome can be accepted in the Boltzmann trial, i.e., $T\infty = 10000$. In the annealing process, the temperature is modulated as $T_{\text{new}} = k \times T_{\text{old}}$, where $k = 0.9$. The number of iterations of the inner loop for a given value of T is set to 5.

The unified synthesis method takes 40 minutes of CPU time on a 1.86 GHz Pentium-M PC with 1G of RAM. The solution thus obtained yields a biochip design with a 9×9 microfluidic array and the completion time for protein assay is 363 s. We illustrate the synthesis results, i.e., assay operation schedule and module placement, using a 3-D box model shown in Fig. 9(a). Each microfluidic module is represented as a 3-D box, the base of which denotes the rectangular area of the module and the height denotes of the time-span of the corresponding assay operation. The projection of a 3-D box on the X-Y plane represents the placement of this module on the microfluidic array, while the projection on the T axis (time axis) represents the schedule of the assay operation. Note that all these boxes are contained in a bin of size $X_{\max} \times Y_{\max} \times T_{\max}$, where $A_{\max} = X_{\max} \times Y_{\max}$; this implies that this design satisfies the specifications of array area and assay completion time. Moreover, there is no overlap between these boxes, thereby avoiding a violation of resource constraints. In addition, the synthesis results also determine the locations of integrated optical detectors. Transparent electrodes for optical detection are used in the microfluidic array. As shown in Fig. 9(b), we can further integrate optical detectors as well as on-chip reservoirs/dispensing ports into the microfluidic array to form a complete digital microfluidic biochip for the protein assay.

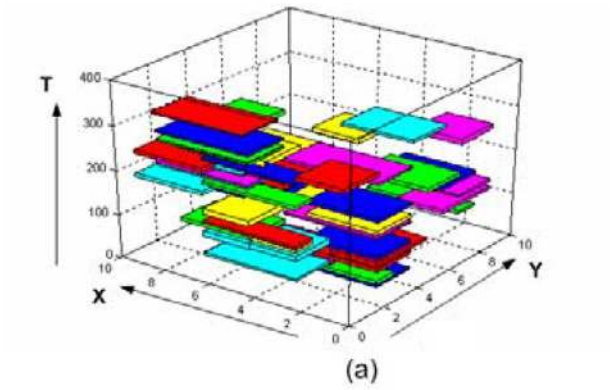
Next we investigate defect tolerant synthesis method using the procedure discussed in Section V. The solution obtained yields a biochip design with a 10×10 microfluidic array and completion time of 377 s, see Fig. 10. Computation time is almost the same with defect-oblivious algorithm.

Although the solution from the enhanced algorithm leads to slightly higher area and assay time than obtained the defect-oblivious design, this design leads to a DTI value of 0.9495, which implies that almost 95% of the modules can be reconfigured if they are affected by defects. This is a considerable improvement over the DTI value of 0.0006 obtained using the defect-aware design method. Note that in the defect-oblivious design, since $\text{DTI} \approx 0$, almost none of the modules can be relocated using partial reconfiguration. The improvement also implies a significant reduction in time-cost for achieving defect tolerance.

The results also show that defect-tolerant synthesis does not necessarily lead to significant time-cost increase. For example, the time-cost just goes only from 363 to 377 s, an increase of less than 3%.

Next, we apply the defect-oblivious and defect-aware synthesis tools to the PCR assay. A total of 34 chromosomes are generated in each population for PCR assay in the PRSA-based algorithm. During evolution, the four best chromosomes are reproduced into the next generation. Six chromosomes in the new population result from the crossover. The remaining 24 chromosomes are obtained from the mutation operators, where eight new chromosomes are from the mutation of genes involved with resource binding, eight from the mutation of genes for scheduling, and eight from the mutation of genes for placement. Here mutation is implemented by randomly generating the new random keys to replace the old ones. Initial temperature $T\infty = 10^4$. In the annealing process, the temperature is modulated as $T_{\text{new}} = k \times T_{\text{old}}$, where $k = 0.9$. The number of iterations of the inner loop for a given value of T is set to three.

Both the two synthesis methods take seven minutes of CPU time on the 1.86-GHz Pentium-M PC with 1 G of RAM. The synthesis results derived for both synthesis methods are shown in the 3-D box in Fig. 11.



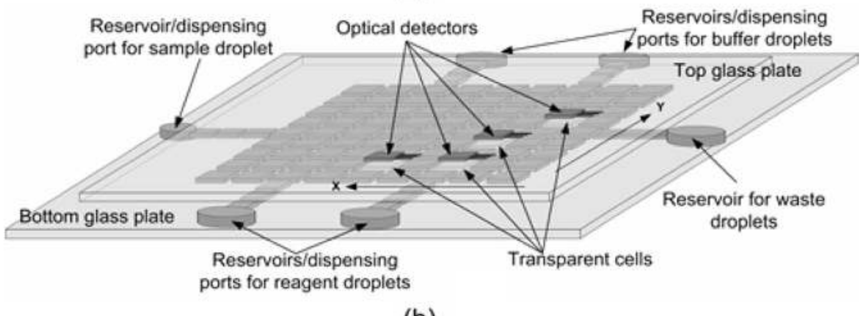

Reservoir/dispensing port for sample droplet
Optical detectors
Reservoirs/dispensing ports for buffer droplets
Top glass plate
Reservoir for waste droplets
Bottom glass plate
Reservoirs/dispensing ports for reagent droplets
Transparent cells
X
Y
(b)

Fig. 9. (a) 3-D model illustrating the synthesis results. (b) Digital microfluidic biochip for a protein assay.

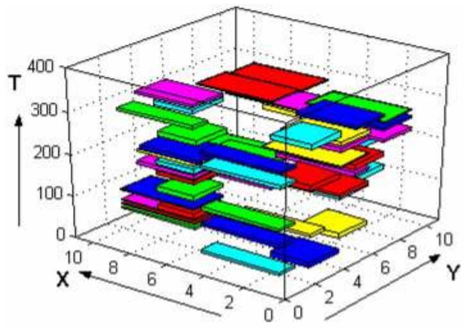


Fig. 10. 3-D model illustrating defect tolerant synthesis results for the protein assay.

B. Defect Tolerance Result From Defect Injection

The defect-oblivious method leads to a biochip design with a 9×7 microfluidic array and an operation time of 16 s while the defect-aware method yields a design with a 9×9 array and an operation time of 18 s.

From the results for both the two assays, i.e., protein assay and PCR mixing stage, we conclude that the defect-aware method usually leads to a slightly larger array area and time-cost compared to the defect-oblivious method. This is reasonable because defect-aware method has to consider reconfiguration *a priori*. Therefore, in ideal situations, i.e., for defect free cases, the defect-oblivious version is a better choice because of the compactness and time efficiency of the design. However, in practice, when defects are likely, the defect-aware methods offers key advantages in many aspects, as highlighted next in Section VI-B.

1) Protein Assay Example: We next evaluate the defect tolerance of the synthesized design by injecting random defects. A design is deemed to be robust if the injected defects can be bypassed by partial reconfiguration. Defects can be classified based on their impact on bioassay functionality.

The first category includes defects that affect only the unused cells in the array. As the biochip functionality is not compromised, these defects are referred to as *benign*. The second category refers to defects that cause significant "fragmentation" of the array, whereby it is no longer possible to relocate a microfluidic module to another part of the array due to lack of availability of defect-free cells. These defects are referred to as *catastrophic*. The third category includes defects that are neither benign nor catastrophic. The microfluidic array can be reconfigured for such defects, hence, we refer to these defects as *repairable*.

A biochip that contains only benign defects is placed in Group I. A biochip that contains catastrophic defects is placed in Group II. Finally, a biochip that contains only repairable and benign defects is placed in Group III. Let N_t be the total number of biochips in a representative sample, and let N_i be the number of biochips in group I, $1 < i < 3$. Clearly $N_1 + N_2 + N_3 = N_t$. We next define two ratios related to the defect tolerance capability of the synthesized biochip: 1) robustness index $r = (N_1 + N_3)/N_t$ and 2) failure index $f = N_2/N_t$.

The goal of defect-aware synthesis is to maximize r and minimize f.

Resynthesis must be carried out for biochips in Group II, i.e., for biochips that suffer from catastrophic defects. Let the bioassay completion time before (after) resynthesis be $T_1(T_2)$.

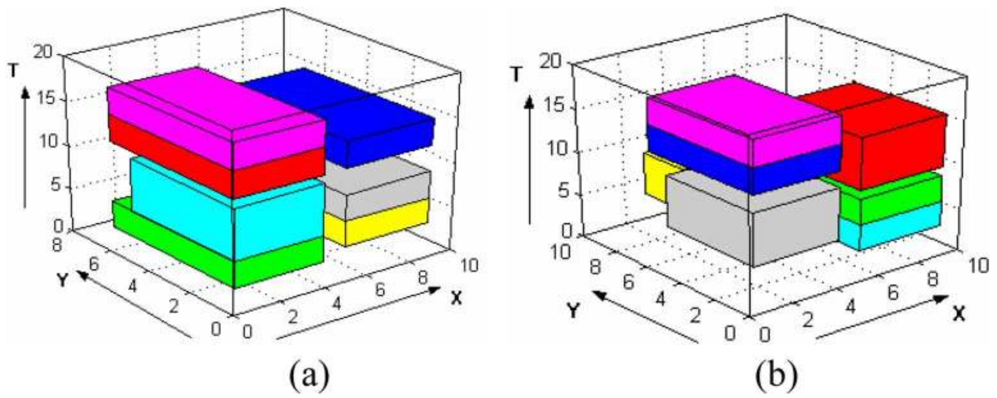


Fig. 11. 3-D model illustrating the synthesis results of a digital microfluidic biochip for PCR mixing stage using (a) defect-oblivious and (b) defect-aware method.

TABLE III
DEFECT TOLERANCE FOR DEFECT-OBLIVIOUS AND DEFECT-AWARE METHOD FOR THE PROTEIN ASSAY. (A) $p = 0.1$ (B) $p = 0.05$ (C) $p = 0.01$

	DTI value	Assay Time (s)	Array Area	*r*	*f*	*td*
Defect-oblivious	0.0003	363	9×9	0.07	0.93	1.45
Defect-aware	0.8954	377	10×10	0.88	0.12	1.14

(a)

	DTI value	Assay Time (s)	Array Area	*r*	*f*	*Td*
Defect-oblivious	0.0006	363	9×9	0.15	0.85	1.34
Defect-aware	0.9495	377	10×10	0.91	0.09	1.08

(b)

	DTI value	Assay Time (s)	Array Area	*r*	*f*	*td*
Defect-oblivious	0.0912	363	9×9	0.30	0.70	1.28
Defect-aware	0.9701	377	10×10	0.97	0.03	1.03

(c)

TABLE IV
DEFECT TOLERANCE FOR DEFECT-OBLIVIOUS AND DEFECT-AWARE METHOD FOR THE PCR ASSAY. (A) $p = 0.1$ (B) $p = 0.05$ (C) $p = 0.01$

	DTI Value	Assay Time (s)	Array Area	*r*	*f*	*td*
Defect-oblivious	0.0045	16	9×7	0.11	0.89	2.64
Defect-aware	0.7634	18	9×9	0.90	0.20	1.93

(a)

	DTI Value	Assay Time (s)	Array Area	*r*	*f*	*td*
Defect-oblivious	0.0136	16	9×7	0.19	0.81	1.92
Defect-aware	0.8921	18	9×9	0.93	0.11	1.31

(b)

	DTI value	Assay Time (s)	Array Area	*r*	*f*	*td*
Defect-oblivious	0.1142	16	9×7	0.28	0.72	1.47
Defect-aware	0.9216	18	9×9	0.98	0.02	1.00

(c)

We define the time degradation td as follow: $td = (T_2 - T_1)/T_1$. Another goal of defect-aware synthesis is to minimize td.

We next take 100 simulated samples of a microfluidic biochip synthesized for the protein assay using both the defect-oblivious and defect-aware method. In each case, we randomly inject defects by assuming that each unit cell is defective with probability p ($p = 1 - q$, $p =$ 0.01, 0.05, 0.1 in our experiments). We then determine the ratios r, f, and td for both methods. The results are shown in Table III. As mentioned in Section VI-A, defect-aware synthesis slightly increases assay time and array area for the protein assay. The key advantage is that it leads to a high DTI value of 0.9495, which implies that almost all modules, once defects are defected, can be reconfigured. This is a significant improvement compared to the DTI value of 0.0006 for defect-oblivious method, where for most modules, defect occurrence is catastrophic and resynthesis has to be carried out.

This improvement is verified by the comparison of failure ratio (f), robustness index (r) and time degradation (td) from Table III.

For all the three value of p, defect-aware synthesis results in a higher value of r and a considerably lower f. Moreover, the defect-aware biochip design also provides a much lower value of td, which implies that for resynthesized biochips, the performance is compromised much less. Since the original time-cost for the two methods are comparable, the difference in td is therefore even more significant. Moreover, td falls more sharply for smaller values of p for the defect-aware synthesis design. Therefore, for low defect probabilities, as is often the cases for mature manufacturing processes, the proposed defect tolerant synthesis method allows resynthesis in the case of catastrophic defects with lower time-cost increase. This feature is often required by many biochip applications.

2) PCR Example: Next, defect injection is carried out for the two biochip designs determined using defect-oblivious and defect-aware methods for the PCR example. Due to the relative lower complexity of the PCR assay, the number of simulation runs is increased to 1000 for smoother averaging of experimental parameters. The results are listed in Table IV.

As in the case of protein assay, a significant enhancement in defect tolerance is observed. In particular, for biochips with high failure probability, e.g., $p = 0.1$, defect-aware synthesis design reduces the failure index from 0.89 to 0.20, an improvement of 78%. Defect-aware synthesis also leads to much lower time degradation. This advantage is even more prominent for biochips with small failure probability. As shown in Table IV-C, defect tolerant synthesis leads to negligible time degradation.

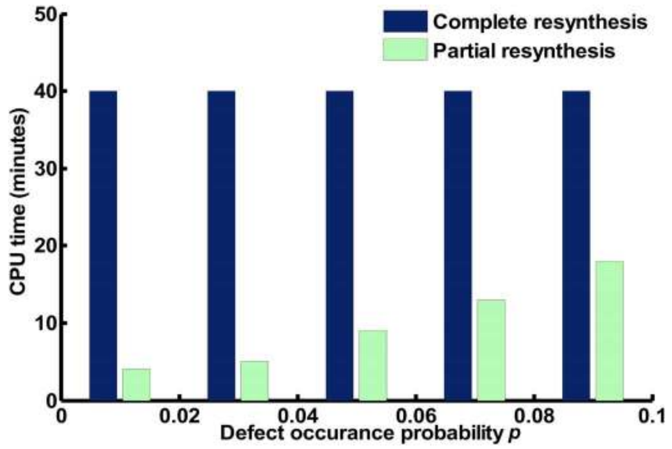


Fig. 12. Computation (CPU) time for partial resynthesis.

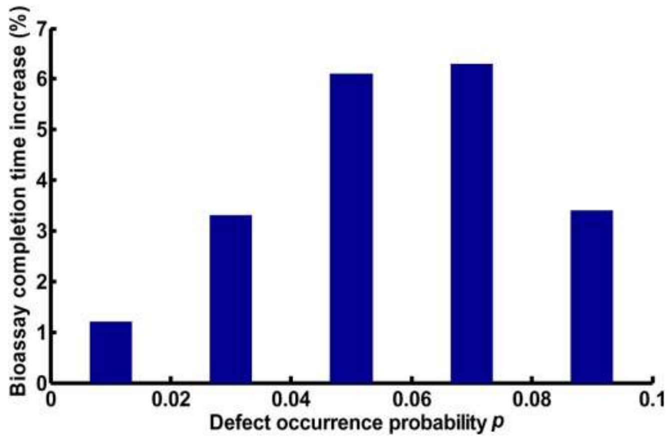


Fig. 13. Percentage increase in bioassay-completion-time (for the protein assay) when partial resynthesis is used instead of complete resynthesis.

3) Evaluation of Partial Resynthesis for Post-Manufacture Defect Tolerance: We carried out partial resynthesis for defect-aware synthesized designs that suffered from catastrophic defects after defect injection. We allowed the protein assay is allowed to be truncated from both ends during the resynthesis. We recorded the computation time needed for resynthesis. The results are shown in Fig. 12. Note that we only present results for protein assay here since the computation time for PCR assay is negligible.

As discussed in Section V-B, partial reconfiguration may result in longer bioassay completion time. Here, for the protein assay, we compare the completion time of the designs obtained using the complete resynthesis method and the proposed partial reconfiguration method. The percentage increase in bioassay-completion-time is defined by

$$\left(\frac{\text{Bioassay completion time from partial resynthesis}}{\text{Bioassay completion time from complete resynthesis}} - 1\right) \times 100\%$$

As expected, partial resynthesis achieves a significant reduction in assay computation time, especially in the case of low defect occurrence probability, e.g., when $p = 0.01$, partial resynthesis saves 90% in CPU time. As a tradeoff, the bioassay completion time is increased only slightly (see Fig. 13). Therefore, it is advantageous to incorporate it in partial resynthesis in the biochip synthesis flow.

VII. Conclusion

We have presented a new defect-aware synthesis method for droplet-based microfluidic biochips. The synthesis procedure, which is based on parallel recombinative simulated annealing, unifies the scheduling of bioassay operations, resource binding, and module placement. We have incorporated both pre and postmanufacture defect tolerance in the unified synthesis method. The real-life example of a protein assay based on the Bradford reaction and a PCR assay have been used to evaluate the effectiveness of the synthesis procedure. This work is expected to facilitate the automated design of biochips. Defect tolerance schemes in the synthesis tool helps improve system reliability for synthesized biochips significantly and efficiently. The biochip user can concentrate on the development of nano- and micro-scale bioassays, leaving implementation details to the synthesis tools. This will in turn pave the way for the integration of biochip components in the next generation of system-on-chip designs, as envisaged by the latest International Technology Roadmap for Semiconductors document.

References

[1] R. B. Fair *et al.*, "Electrowetting-based on-chip sample processing for integrated microfluidics," in *Proc. IEDM*, 2003, pp. 32.5.1–32.5.4.
[2] E. Verpoorte and N. F. De Rooij, "Microfluidics meets MEMS," *Proc. IEEE*, vol. 91, no. 6, pp. 930–953, Jun. 2003.
[3] J. Zeng and T. Korsmeyer, "Principles of droplet electrohydrodynamics for lab-on-a-chip," *Lab on a Chip*, vol. 4, pp. 265–277, 2004.
[4] D. Chatterjee *et al.*, "Droplet-Based microfluidics with nonaqueous solvents and solutions," *Lab on a Chip*, vol. 6, pp. 199–206, 2006.
[5] S. K. Cho *et al.*, "Creating, transporting, cutting, and merging liquid droplets by electrowetting-based actuation for digital microfluidic circuits," *J. Microelectromech. Syst.*, vol. 12, pp. 70–80, 2003.
[6] P. S. Dittrich, K. Tachikawa, and A. Manz, "Micro total analysis systems: Latest advancements and trends," *Anal. Chem.*, vol. 78, pp. 3887–3908, 2006.
[7] T. H. Schulte *et al.*, "Microfluidic technologies in clinical diagnostics," *Clinica Chimica Acta*, vol. 321, pp. 1–10, 2002.
[8] F. Su *et al.*, "Microfluidics-based biochips: Technology issues, implementation platforms, and design automation challenges," *IEEE Trans. Comput.-Aided Des. Integr. Circuits*, vol. 25, no. 1, pp. 211–223, Feb. 2006.
[9] K. Chakrabarty and J. Zeng, "Design automation for microfluidics-based biochips," *ACM J. Emerging Technol. in Comput. Syst.*, vol. 1, pp. 186–223, Dec. 2005.
[10] R. B. Fair *et al.*, "Chemical and biological applications of digital microfluidic devices," *IEEE Design Test Comput.*, vol. 24, no. 1, pp. 10–24, Jan. 2006.
[11] Silicon Biosystems (2008). [Online]. Available: http://www.silicon-biosystems.com
[12] P.-H. Yuh *et al.*, "Placement of digital microfluidic biochips using the T-tree formulation," in *Proc. DAC*, 2006, pp. 931–934.
[13] F. Su and K. Chakrabarty, "Architectural-level synthesis of digital microfluidics-based biochips," in *Proc. ICCAD*, 2004, pp. 223–228.
[14] F. Su and K. Chakrabarty, "Unified high-level synthesis and module placement for defect-tolerant microfluidic biochips," *Proc. DAC*, pp. 825–830, 2005.
[15] G. Li and N. Aluru, "Efficient mixed-domain analysis of electrostatic MEMS," *IEEE Trans. Comput.-Aided Des. Integr. Circuits*, vol. 22, pp. 1228–1242, 2003.
[16] T. Mukherjee and G. K. Fedder, "Design methodology for mixed-domain systems-on-a-chip [MEMS design]," in *Proc. IEEE VLSI Syst. Level Des.*, 1998, pp. 96–101.
[17] S. K. Tewksbury, "Challenges facing practical DFT for MEMS," in *Proc. Defect Tolerance VLSI Syst.*, 2001, pp. 11–17.
[18] T. Zhang *et al., Microelectrofluidic Systems: Modeling and Simulation*. Boca Raton, FL: CRC, 2002.

[19] W. E. Dougherty and D. E. Thomas, "Unifying behavioral synthesis and physical design," in *Proc. DAC*, 2000, pp. 756–760.

[20] K. Bazargan *et al.*, "Integrating scheduling and physical design into a coherent compilation cycle for reconfigurable computing architectures," in *Proc. DFAC*, 2001, pp. 635–640.

[21] V. Srinivasan *et al.*, "Protein stamping for MALDI mass spectrometry using an electrowetting-based microfluidic platform," in *Proc. SPIE*, 2004, vol. 5591, pp. 26–32.

[22] M. Garey and D. Johnson*, Computers and Intractability-a Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[23] D. F. Wong, H. W. Leong, and H. W. Liu*, Simulated Annealing for VLSI Design*. New York: Springer-Verlag, 1988.

[24] K. Kurbel, B. Schneider, and K. Singh, "Solving optimization problems by parallel recombinative simulated annealing on a parallel computer-an application to standard cell placement in VLSI design," *IEEE Trans. Syst., Man Cybern., B*, vol. 28, no. 3, pp. 454–461, Mar. 1998.

[25] S. W. Mahfoud and D. E. Goldberg, "Parallel recombinative simulated annealing: A genetic algorithm," *Parallel Comput.*, vol. 21, pp. 1–28, 1995.

[26] J. C. Bean, "Genetics and random keys for sequencing and optimization," *ORSA J. Comput.*, vol. 6, pp. 154–160, 1994.

[27] G. De Micheli*, Synthesis and Optimization of Digital Circuits*. New York: McGraw-Hill, 1994.

[28] K. Bazargan and M. Sarrafzadeh, "Fast template online placement for reconfigurable computing systems," in *Proc. IEEE Symp. Field-Programmable Custom Computing Mach.*, 1999, pp. 300–302.

[29] F. Su *et al.*, "Defect-oriented testing and diagnosis of digital microfluidics-based biochips," in *Proc. IEEE ITC*, 2005, pp. 487–496.

[30] T. Xu and K. Chakrabarty, "Parallel scan-like testing and fault diagnosis techniques for digital microfluidic biochips," in *Proc. IEEE ETS*, 2007, pp. 63–68.

[31] F. Su and K. Chakrabarty, "Module placement for fault-tolerant microfluidics-based biochips," *ACM Trans. Design Autom. Electron. Syst.*, vol. 11, pp. 682–710, 2006.

[32] C. G. J. Schabmueller *et al.*, "Closed chamber PCR chips for DNA amplification," *Eng. Sci. Educ. J.*, vol. 9, no. 6, pp. 259–264, 2000.

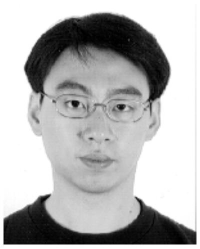
Tao Xu (S'07) received the B.E. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2005 and the M.S. degree in electrical and computer engineering from Duke University, Durham, NC, in 2007, where he is pursuing the Ph.D. degree in electrical and computer engineering.

His research interests include design and testing of mixed-technology microsystems, electronic design automation, mixed-signal VLSI design, MEMS modeling and simulation.

Krishnendu Chakrabarty (S'92–M'96–SM'00–F'08) received the B.Tech. degree from the Indian Institute of Technology, Kharagpur, India, in 1990, and the M.S.E. and Ph.D. degrees from the University of Michigan, Ann Arbor, in 1992 and 1995, respectively, all in computer science and engineering.

He is currently Professor of Electrical and Computer Engineering at Duke University. His current research projects include: testing and testing and design-for-testability of system-on-chip integrated circuits; digital microfluidic biochips; logic circuits based on DNA self-assembly; delay-tolerant wireless networks. Prof. Chakrabarty has authored five books—*Microelectrofluidic Systems: Modeling and Simulation* (CRC Press, 2002), *Test Resource Partitioning for System-on-a-Chip* (Kluwer, 2002), *Scalable Infrastructure for Distributed Sensor Networks* (Springer, 2005), *Digital Microfluidics Biochips: Synthesis, Testing, and Reconfigutaion Techniques* (CRC Press, 2006), and *Adaptive Cooling of Integrated Circuits using Digital Microfluidics* (Artech House, April 2007)—and edited the book volumes SOC *(System-on-a-Chip) Testing for Plug and Play Test Automation* (Kluwer, 2002) and *Design Automation Methods and Tools for Microfluidics-Based Biochips* (Springer, 2006). He has contributed over a dozen invited chapters to book volumes, published 260 papers in archival journals and refereed conference proceedings, and delivered over 100 keynote, plenary, and invited talks. He holds a U.S. patent in built-in self-test and is a co-inventor of a pending US patent on sensor networks.

Prof. Chakrabarty served as a Distinguished Visitor of the IEEE Computer Society for 2005–2007 and as a Distinguished Lecturer of the IEEE Circuits and Systems Society for 2006–2007. He is an Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, IEEE TRANSACTIONS ON VLSI SYSTEMS, IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS, ACM JOURNAL ON EMERGING TECHNOLOGIES IN COMPUTING SYSTEMS, an Editor of *IEEE Design and Test of Computers*, and an Editor of *Journal of Electronic Testing: Theory and Applications (JETTA)*. He served as Program Chair for the IEEE Asian Test Symposium in 2005 and the CAD, Design, and Test Conference for the 2007 IEEE Symposium on Design, Integration, Test, and Packaging of MEMS/MOEMS. He is a recipient of the National Science Foundation Early Faculty (CAREER) award and the Office of Naval Research Young Investigator award. He is a recipient of best paper awards at the 2007 IEEE International Conference on VLSI Design, the 2005 IEEE International Conference on Computer Design, and the 2001 IEEE Design, Automation and Test in Europe (DATE) Conference. He is also a recipient of the Humboldt Research Fellowship, awarded by the Alexander von Humboldt Foundation, Germany, and the Mercator Visiting Professorship, awarded by the Deutsche Forschungsgemeinschaft, Germany. He is a recipient of Duke University's 2008 Dean's Award for Excellence in Mentoring. He is a Senior Member of ACM, and a member of Sigma Xi.



Fei Su received the B.S. degree in automation, and the M.S. degree in detection and automation devices from Tsinghua University, Beijing, China, in 1999 and 2001, respectively, and the M.S. and Ph.D. degrees in electrical and computer engineering from Duke University, Durham, NC, in 2003 and 2006, respectively.

He is currently a Senior DFT (Design-for-Test) Engineer of Intel Corporation, Folsom, CA. His research interests include DFT, testing and CAD for SOC and mixed microsystems (e.g., microfluidic biochips), mixed signal circuits and high-speed IO design. He is a coauthor of *Digital Microfluidic Biochip: Synthesis, Testing and Reconfiguration Techniques* (CRC Press, 2006).

Dr. Su is a recipient of European Design Automation Association (EDAA) Outstanding Dissertation Award 2006 and the Best Paper Award at the 2007 IEEE International Conference on VLSI Design. He is a member of IEEE TTTC (Test Technology Technical Council) and ACM SIGDA (Special Interest Group in Design Automation). He serves as an invited reviewer of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN, IEEE *Design and Test of Computers*, *ACM Journal on Emerging Technologies in Computing Systems*, and IEEE/ACM Design Automation Conference.