Integrated Control-Path Design and Error Recovery in the Synthesis of Digital Microfluidic Lab-on-Chip

YANG ZHAO, TAO XU, and KRISHNENDU CHAKRABARTY
Duke University

Recent advances in digital microfluidics have led to tremendous interest in miniaturized lab-on-chip devices for biochemical analysis. Synthesis tools have also emerged for the automated design of lab-on-chip from the specifications of laboratory protocols. However, none of these tools consider control flow or address the problem of recovering from fluidic errors that can occur during on-chip bioassay execution. We present a synthesis method that incorporates control paths and an error-recovery mechanism in the design of a digital microfluidic lab-on-chip. Based on error-propagation estimates, we determine the best locations for fluidic checkpoints during biochip synthesis. A microcontroller coordinates the implementation of the control-flow-based bioassay by intercepting the synthesis results that are mapped to the software programs. Real-life bioassay applications are used as case studies to evaluate the proposed design method. For a representative protein assay, compared to a baseline chip design, the biochip with a control path can reduce the completion time by 30% when errors occur during the implementation of the bioassay.

Categories and Subject Descriptors: B.7.2 [**Integrated Circuits**]: Design Aids—*Placement and routing*; B.7.3 [**Integrated Circuits**]: Reliability and Testing—*Error checking*; J.3 [**Life and Medical Sciences**]: Biology and genetics, Health

General Terms: Algorithms, Design, Performance, Reliability

Additional Key Words and Phrases: Error recovery, synthesis, microfluidics, biochips

ACM Reference Format:

Zhao, Y., Xu, T., and Chakrabarty, K. 2010. Integrated control-path design and error recovery in the synthesis of digital microfluidic lab-on-chip. ACM J. Emerg. Technol. Comput. Syst. 6, 3, Article 11 (August 2010), 28 pages.
DOI = 10.1145/1777401.1777404 http://doi.acm.org/10.1145/1777401.1777404

This research was supported in part by the National Science Foundation under grants CCF-0541055 and CCF-0914895.

Authors' addresses: Y. Zhao, K. Chakrabarty, Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708; email: yz61@ee.duke.edu; krish@ee.duke.edu; T. Xu, Cisco Systems, 7025 Kit Creek Road, Research Triangle Park, NC 27560; email: txu2@cisco.com.

DOI 10.1145/1777401.1777404 http://doi.acm.org/10.1145/1777401.1777404

1. INTRODUCTION

Digital microfluidics is an emerging technology that provides fluid-handling capability on a chip [Fair et al. 2007]. By reducing the rate of sample and reagent consumption, digital microfluidics lab-on-chip enables on-chip immunoassays, point-of-care clinical diagnostics, and high-throughput DNA sequencing. Bioassay protocols are run on a lab-on-chip through the manipulation of discrete droplets of nanoliter volume under clock control on a two-dimensional electrode array of electrodes ("unit cells"). A unit cell in the array includes a pair of electrodes that acts as two parallel plates. The bottom plate contains a patterned array of individually controlled electrodes, and the top plate is coated with a continuous ground electrode. A droplet rests on a hydrophobic surface over an electrode, as shown in Figure 1.

Droplets are moved by applying a control voltage to a unit cell adjacent to the droplet and, at the same time, deactivating the one under the droplet. This electronic method of wettability control creates interfacial tension gradients that move the droplets to the charged electrode. Fluid-handling operations such as droplet merging, splitting, mixing, and dispensing can be executed in a similar manner. Droplet routes and operation schedules are programmed into a microcontroller that drives the electrodes.

Several complex biomedical procedures have recently been demonstrated on the digital microfluidics platform, for example, gene sequencing through synthesis [Fair et al. 2007], protein crystallization for drug discovery [Xu et al. 2008], and cell sorting [Medoro 2007]. These advances in technology and applications serve as a powerful driver for research on computer-aided design (CAD) tools for lab-on-chip design. A number of CAD methods, including techniques for scheduling, module placement, and droplet routing, have been developed for the design and use of microfluidic lab-on-chip [Chakrabarty and Su 2006; Chakrabarty and Zeng 2006, 2005; Maftei et al. 2008; Bohringer 2006; Cho and Pan 2008; Griffith et al. 2006; Su and Chakrabarty 2005, 2008; Su et al. 2006; Ricketts et al. 2006; Xu and Chakrabarty 2008; Yuh et al. 2007b, 2007a].

For biomedical applications such as clinical diagnostics, it is necessary to verify the correctness of on-chip fluidic operations. The status of a bioassay can be monitored by examining parameters such as the volume of the product droplet, sample concentration in the product, and detector readout. If an error occurs during the execution of the bioassay, for example, an intermediate product droplet exceeds the normal volume, the assay outcomes can be misleading. Therefore, it is important to detect such errors as early as possible and re-execute the fluidic operations to obtain correct assay outcomes. However, current synthesis tools only provide a "data path" implementation involving the scheduling and placement of microfluidic modules. None of these tools consider control flow or address the problem of recovering from fluidic errors that can occur during an on-chip bioassay. Fluidic operations in these designs are carried out following the pre-determined schedule without any feedback. If an error occurs during the execution of the assay, it can only be detected when the assay is completed, and then the entire bioassay must be repeated.

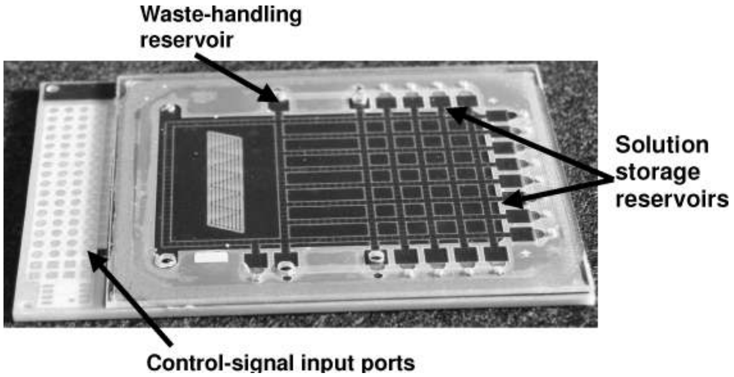

Fig. 1. A fabricated and packaged digital microfluidic array [Advanced Liquid Logic].

Such repetitive executions lead to wastage of samples and reagents, and an undue increase in the assay time. Therefore, it is necessary to monitor intermediate results and design a feedback control mechanism during bioassay execution to repeat only the fragment of the bioassay where errors are detected. While the notion of error recovery is taken here from reliable computing, a key difference from computing is that the rollback strategy must take into account the availability of sample volumes and intermediate solutions of appropriate concentrations. Rollback can only be made to points where droplets of required concentrations have been stored.

In this article, we propose a synthesis method that incorporates a control path in the design of a digital microfluidic lab-on-chip. We describe the implementation of the checkpoint and the corresponding re-execution (rollback) subroutine to correct the detected error. Based on the concept of error-propagation estimates, we determine the best locations for fluidic checkpoints during biochip synthesis. The synthesis results are mapped to the software programs and stored in a microcontroller. We use the microcontroller to coordinate the implementation of the control-flow-based bioassay by intercepting the software programs. Real-life bioassay applications are used as case studies to evaluate the proposed design method.

The remainder of the article is organized as follows. In Section 2, we discuss related prior work on fault models, automated synthesis tools for digital microfluidic lab-on-chip, and the on-chip integration of detection mechanisms. Section 3 describes an efficient control-path design method based on the concept of checkpoints and the corresponding re-execution subroutines. Section 4 presents the proposed checkpoint-insertion method based on the concept of error-propagation estimates. In Section 5, we describe the simultaneous incorporation of control paths and synthesis of the biochip. In Section 6, we describe how a microcontroller is used to coordinate the implementation of the control-flow-based bioassay by intercepting the software programs resulting from chip synthesis. In Section 7, a large-scale protein assay, an interpolating mixing architecture, and synthetic bioassays are used as case studies to evaluate the proposed design method. Finally, conclusions are drawn in Section 8.

2. RELATED PRIOR WORK

Digital microfluidic lab-on-chip synthesis can be conceptually viewed as consisting of two stages, namely architectural synthesis and physical design. Architectural synthesis maps a behavioral model (e.g., a sequencing graph) for a bioassay to a macroscopic structure of the lab-on-chip, which specifies scheduling and resource binding. A system design method based on classical high-level synthesis is described in Su and Chakrabarty [2008].

Physical design generates the lab-on-chip layout, including the placement of microfluidic modules such as mixers, storage units, detectors, as well as droplet-routing pathways. In Yuh et al. [2007b], the placement problem of digital microfluidic biochips is modeled as a temporal (3D) floorplanning problem, in order to simultaneously perform scheduling and physical placement. In order to ensure defect tolerance and perform a bioassay on a biochip with defective sites, the proposed placement algorithm models each defective cell as an obstacle and does not allow overlaps between operations and obstacles.

A synthesis method based on parallel recombinative simulated annealing (PRSA) [Mahfoud and Goldberg 1995] is presented in Su and Chakrabarty [2005]. First, the different bioassay operations (e.g., mixing and dilution), and their mutual dependences are represented using a sequencing graph. Next, a combination of simulated annealing and genetic algorithms are used for unified resource binding, operation scheduling, and module placement. Each candidate solution, that is, a design point, is represented using a chromosome. In each chromosome, operations are randomly bound to resources. Based on the binding results, list scheduling is used to determine the start times of operations, that is, each operation starts with a random latency after its scheduled time. Finally, a module placement is derived based on the resource binding and scheduling information. A weighted sum of area and time cost is used to evaluate the quality of the design. The design is improved through a series of genetic evolutions based on PRSA.

As in the case of integrated circuits (ICs), an increase in the density and area of microfluidics-based biochips will lead to high defect densities, thereby reducing yield, especially for newer technologies. Moreover, since complicated fluidic operations are repeatedly executed in compact microfluidic arrays, a group of cells is repeatedly required to perform a large number of operations, hence malfunctions can occur for certain patterns of droplet movement or fluidic operations [Xu and Chakrabarty 2007]. A comprehensive set of fault models for defects and malfunctions is described in Table I.

For example, when electrodes are actuated for excessive duration, irreversible charge will concentrate on the dispensing electrode. This leads to the fault “dispensing-stuck-on” during the dispensing operation, that is, a droplet is dispensed but not fully cut off from the reservoir. Thereby, no droplet can be dispensed from the reservoir. Furthermore, due to the electrode electrostatic property variation in fabrication, unequal actuation voltages can be applied to electrodes where a droplet is undergoing the splitting operation. This results in two outcome droplets with unbalanced volumes. In the protein assay, if the droplet with incorrect volume dilutes with the other droplet with normal

Table I. Fault Models for Digital Microfluidic Biochips [Xu and Chakrabarty 2007]

Cause of Defect	Defect Type	Fault Model	Observable Error
Excessive actuation voltage applied to an electrode	Dielectric breakdown	Droplet-electrode short (a short between the droplet and the electrode)	Droplet undergoes electrolysis, which prevents its further transportation
Electrode actuation for excessive duration	Irreversible charge concentration on an electrode	Electrode-stuck-on (the electrode remains constantly activated)	Unintentional droplet operations or stuck droplets
Excessive mechanical force applied to the chip	Misalignment of parallel plates (electrodes and ground plane)	Pressure gradient (net static pressure in some direction)	Droplet transportation without activation voltage
Coating failure	Non-uniform dielectric layer	Dielectric islands (islands of Teflon coating)	Fragmentation of droplets and their motion is prevented
Abnormal metal layer deposition and etch variation during fabrication	Grounding Failure	Floating droplets (droplet are not anchored)	Failure of droplet transportation
	Broken wire to control source	Electrode open (electrode actuation is not possible)	Failure to activate the electrode for droplet transportation
	Metal connection between two adjacent electrodes	Electrode short (short between electrodes)	A droplet resides in the middle of the two shorted electrodes, and its transport along one or more directions cannot be achieved
Particle contamination or liquid residue	A particle that connect two adjacent electrodes	Electrode short	
Protein adsorption	Sample residue on electrode surface	Resistive open at electrode	Droplet transportation is impeded
		Contamination	Assay results are outside the range of possible outcomes
Cause of Malfunction	Malfunction Type	Fault Model	Observable Error
Electrode actuation for excessive duration	Irreversible charge concentration on the dispensing electrode	Dispensing-stuck-on (droplet is dispensed but not fully cut off from the reservoir)	No droplet can be dispensed from the reservoir
Electrode shape variation in fabrication	Deformity of electrodes	No overlap between droplets to be mixed and center electrode	Mixing failure
Electrode electrostatic property variation in fabrication	Unequal actuation voltages	Pressure gradient (net static pressure in some direction)	Unbalanced volumes of split droplets
Bad soldering	Parasitic capacitance in the capacitive sensing circuit	Oversensitive or insensitive capacitive sensing	False positive/ negative in detection

volume, the outcome droplet will have incorrect volume and its concentration will be different from the desired one. In this manner, the error in droplet volume and concentration will be propagated along the dilution tree until the assay's endpoint is reached. Therefore, the concentration at the end of the assay will be different from the desired concentration.

However, prior synthesis methods suffer from the drawback that they do not implement any control flow to recover from errors that can occur during bioassay execution. The fluidic operations are carried out following the predetermined schedule without any sensor feedback. If an error is detected at the end of the assay, the entire bioassay must be repeated. As a result, a monitoring and an appropriate feedback-control mechanism must be implemented during bioassay execution. A monitoring method can determine the status of the assay and the quality of intermediate products using on-chip detectors at several checkpoints. If a malfunction is detected or the quality of an intermediate product fails to meet predetermined requirements, only a fragment of the bioassay must be re-executed. Here we refer to the monitoring and control mechanisms as the "control path" for the digital microfluidic lab-on-chip.

The quality of an intermediate product droplet can be determined by examining the analyte concentration level. The concentration variation can be revealed by the color change in the product droplet. Photodetectors are capable of detecting color changes by converting the light from a light emitting diode (LED) into either current or voltage. Recent work has demonstrated the feasibility of integrating photodetectors in the microfluidic array [Luan et al. 2008; Minas et al. 2005; Srinivasan et al. 2003].

An InGaAs-based thin-film inverted metal-semiconductor-metal (I-MSM) photodetector has been reported in Cho et al. [2002]. The I-MSM photodetectors are independently fabricated and subsequently bonded to the metal contact pads on the Benzocyclobutene (BCB)/SiO_2/Si substrate. To carry out chip-level integration, a 3-μm layer of SiO_2 is deposited on the Si substrate using plasma enhanced chemical vapor deposition (PECVD), followed by a spin-coated 1-μm-thick BCB core layer. The 3-μm layer of SiO_2 acts as a cladding and buffer layer for the BCB layer. Next, the thin-film photodetectors are separately fabricated and bonded to the pads on the Si substrate. The detection area of this photodetector is 100 μm × 150 μm.

In Seo et al. [2002], GaN thin-film metal-semiconductor-metal (MSM) photodetectors are heterogeneously integrated onto a host substrate of SiO_2-Si. GaN epitaxial layers are grown on lithium gallate ($LiGaO_2$) substrates, which leads to small lattice mismatch. The doped GaN MSM photodetector on $LiGaO_2$ is grown using plasma-assisted radio frequency molecular beam epitaxy. The MSM photodetectors used for all measurements were 47 μm long, with 2-μm finger width and 5-μm finger spacing, and a detection area of 50 μm × 50 μm.

In Srinivasan et al. [2003], an optical detection system is integrated with the digital microfluidic array. It is set up perpendicular to the main plain of the microfluidic array. It consists of a light emitting diode (LED) and a photodiode, which is a light-to-voltage converter. The sample concentration can be measured from the absorbance of the products using a rate kinetic method [Srinivasan et al. 2004]. The availability of the detectors described above provides the motivation for fluidic rollback schemes based on error detection.

3. CONTROL-PATH DESIGN AND ROLLBACK-RECOVERY MECHANISM

3.1 Checkpointing and Re-Execution Subroutine

Given a bioassay sequencing graph, control-path design first determines which operations need to be monitored. A checkpoint is inserted at the output of the fluidic operation that needs to be monitored. A *checkpoint* is defined as the storage of the intermediate product droplet at the output of the fluidic operation. This droplet is stored at an on-chip storage unit.

Error detection is performed for the intermediate product droplet at the checkpoint. The droplet is transported to an on-chip photodetector or a capacitive-sensing circuit. For error detection using photodetectors, the droplet is subjected to a detection operation to determine whether there is an error, that is, whether the concentration of the droplet is outside the acceptable range. Note that if the intermediate product droplet is transparent, it has to be mixed with reagent droplets to generate a non-transparent (colored) analyte droplet. This analyte droplet can be examined colorimetrically by photodetectors. For example, the colorimetric detection of sulfate is described in Madsen and Murphy [1981]. The sulfate concentration is determined indirectly based on the competitive reaction of sulfate and Methylthymol blue (MTB) with barium in solution based on absorbance measurements of either uncomplexed MTB or the MTB-barium complex. In this case, the product droplet used for error detection is typically not available for subsequent operations. For detection using a capacitive-sensing circuit [Pollack 2001], the intermediate product droplet is subjected to a volumetric test to determine whether the volume of the droplet is outside the acceptable range. When a droplet undergoes capacitive-sensing detection, it can be utilized for subsequent operations.

If no error is detected, the intermediate product droplet can be stored and utilized for subsequent operations. If an error is detected, the intermediate product droplet is transported to the waste reservoir, and the detection mechanism will trigger rollback recovery to correct the error. For each checkpoint, control-path design determines the corresponding *re-execution subroutine* for rollback recovery. The re-execution subroutine includes all the fluidic operations from the immediate upstream checkpoint along all the paths in the sequencing graph model to the current checkpoint. Note that during bioassay execution, a checkpoint can only be reached when no failure is detected in all its upstream checkpoints. This implies that if an error is detected, it must be localized among the operations between the current checkpoint and the immediate upstream checkpoint. Therefore, by re-executing the subroutine, the error can be corrected.

Figure 2 shows the insertion of checkpoints and the corresponding re-execution subroutines. Three checkpoints C_1, C_2 and C_3 are inserted at the outputs of fluidic operations O_0, O_2 and O_5, respectively. A re-execution subroutine is assigned to checkpoint C_2. This subroutine includes operations O_1 and O_2. The outputs of operations O_0 and O_5 feed the inputs of fluidic operations in the subroutine. If an error is detected for the intermediate product droplet at checkpoint C_2 while no error is detected at other checkpoints as shown in

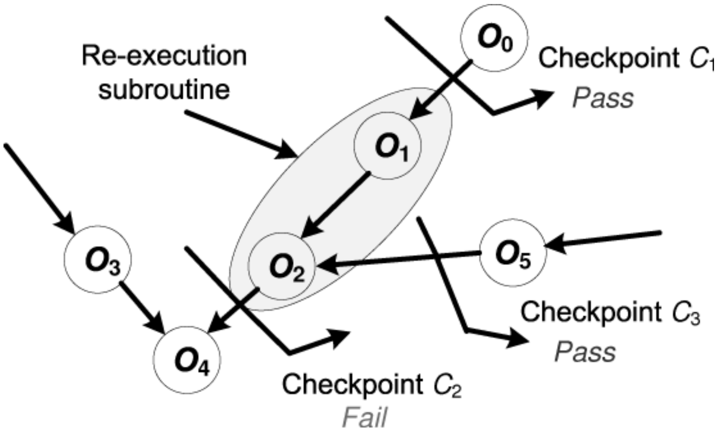

Fig. 2. Checkpoint insertion and re-execution subroutine.

Figure 2, this implies that the error is localized among the operations between C_1 and C_2. In this case, the detection mechanism will trigger the subroutine, and operations O_1 and O_2 will be re-executed to correct the error.

3.2 Droplet Preparation for the Re-Execution Subroutine

The re-execution of a subroutine consumes additional droplets. For example, in Figure 2, suppose an error is detected at checkpoint C_2. To implement the corresponding subroutine to correct the error, we also need input droplets from operations whose outputs feed the inputs of operations in the subroutine (i.e., O_0 and O_5). These droplets should have been generated or stored on-chip, in order to be retrieved during the implementation of the subroutine of checkpoint C_2. They are referred to as “copy droplets”. Here we describe how these additional input droplets are obtained.

Assume that O_0 or O_5 is a dilution operation. After the implementation of O_0 or O_5 during the bioassay, there are two 1x product droplets. Since there is a checkpoint at the output of O_0 or O_5, one product droplet will be transported to a photodetector or a capacitive-sensing circuit for error detection. Note that if the product droplet is transparent, we use the capacitive-sensing circuits for error detection, to ensure that the droplet can be used for subsequent operation O_1 or O_2 during the bioassay. The other product droplet is transported to an on-chip storage unit as the copy droplet for O_0 or O_5, so that it can be retrieved during the re-execution of the subroutine for checkpoint C_2.

Next, assume that O_0 or O_5 is a mix operation, where two 1x input droplets are mixed together into a 2x product droplet. In this case, we insert a split operation after O_0 or O_5, in order to split the 2x product droplet into two 1x product droplets. One product droplet is used for the bioassay, while the other is stored as a copy droplet, so that it can be retrieved during the re-execution of the subroutine for C_2.

Next, suppose that O_0 or O_5 is a dispensing operation. Before the re-execution of the subroutine for C_2, we can always implement O_0 or O_5 to dispense a droplet and use it to feed the inputs of the re-execution subroutine. Therefore, we do not need any copy droplet for O_0 or O_5.

Finally, assume that O_0 or O_5 is a split operation. During the bioassay, one product droplet of O_0 or O_5 is consumed by subsequent operation O_1 or O_2, while the other product droplet is consumed by other subsequent operations. Therefore, no product droplet of O_0 or O_5 is stored as a copy droplet. In this case, we include O_0 and O_5 into the current subroutine for checkpoint C_2 to form a new subroutine that includes O_0, O_1, O_2 and O_5. We also determine whether we can obtain copy droplets from operations whose outputs feed the inputs of operations in the new subroutine, that is, predecessor operations of O_0 and O_5. If copy droplets exist at the outputs of the predecessor operations, we can use these copy droplets to re-execute the new subroutine (including O_0, O_1, O_2 and O_5) for checkpoint C_2. Otherwise, we continue the above process to enlarge the subroutine for checkpoint C_2 by including predecessor operations, until we reach predecessor operations that can provide copy droplets to feed the inputs of operations in the subroutine.

3.3 Time Cost

The time cost for implementing a checkpoint and its corresponding re-execution subroutine can be divided into four parts. The first part is the time cost for the storage of the intermediate product droplet at the checkpoint. When the intermediate product droplet is generated at the output of the fluidic operation, it will be transported to the on-chip detector immediately at the next clock cycle. Therefore, no time cost is incurred for this step.

The second part is the time cost for transporting the intermediate product droplet to an on-chip detector. Since droplet movement on a digital microfluidic array is fast (e.g., typically 8 Hz frequency [Pollack 2001]) compared to other fluidic operations (e.g., typically at 1 Hz), we can ignore the droplet-movement time for checkpointing.

The third part, that is, the time needed for error detection of the intermediate product droplet, depends on the detection mechanism. For example, the time for error detection using an LED-photodiode detector is typically 5 seconds [Pollack 2001]. The capacitive-sensing circuit operates at relatively high frequency (15 kHz) [Pollack 2001]. In contrast, in traditional digital circuits, the intermediate output signal is sent to control hardware for error detection. The signal is compared with a signature stored in the controller to determine whether it is erroneous. The duration of the detection process depends on the clock frequency of the controller (usually in hundreds of MHz), hence the detection time for digital circuits is negligible compared to digital microfluidics.

The fourth part is the time cost for implementing the re-execution subroutine associated with the checkpoint. It can be decomposed into two subparts. The first subpart is the time needed to retrieve stored copy droplets and bring them to the inputs of the fluidic operations in the subroutine. It can be calculated based on the maximum distance between their on-chip storage units and the modules where the fluidic operations in the subroutine are implemented. The second subpart is the time needed to re-execute the subroutine, for example, operations O_1 and O_2 for checkpoint C_2 in Figure 2. It can be calculated as the duration from the start time of O_1 to the end time of O_2 in the bioassay schedule. In traditional digital circuits, the detected error can be corrected by re-executing the logic operations between two checkpoints in the circuit. Since the digital circuit usually works at high clock frequency, the absolute (not relative) time cost for the re-execution is not excessive. In a digital microfluidic biochip, since fluidic operations are usually implemented at low frequency (a few Hz or tens of Hz), the time cost for the re-execution is much higher than that for a digital circuit.

3.4 Space Cost

Two types of fluidic device resources are used for the checkpoint and the corresponding re-execution subroutine. One type is the reconfigurable resource, which includes the storage units for the intermediate product droplets and copy droplets. For example, in Figure 2, storage units should be used to store the intermediate droplets of operations O_0, O_2, and O_5, as well as the copy droplets of operations O_0, O_2, and O_5. The reconfigurable resource can be dynamically created using available electrodes in the microfluidic array during the target bioassay, hence it does not introduce any space cost.

The other type is the non-reconfigurable resource, which includes the on-chip photodetectors. Since there are more detection operations in the bioassay with control paths than that without control paths, additional photodetectors have to be added. The introduced space cost depends on the area that the photodetectors occupy on the microfluidic array. For example, a photodiode detector is set up perpendicular on one electrode to the microfluidic array, and the adjacent eight electrodes are used as the guard ring in order to avoid inadvertent mixing. The number of photodetectors also affects the time needed for rollback recovery. With these additional detectors, several intermediate product droplets can be detected concurrently on different detectors, which reduces the error-detection time.

In contrast, in a digital circuit, since all the circuit modules have been fabricated and the error information is transported using electrical signals, there is no space cost for error recovery.

4. ERROR-PROPAGATION ESTIMATES FOR CHECKPOINT INSERTION

Although checkpoint monitoring and rollback recovery mechanism are useful for recovering from fluidic errors that can occur during on-chip bioassay execution, they lead to increased assay completion times. Therefore, careful design is needed to limit the number of checkpoints and the size of the re-execution segment for each checkpoint. We propose an efficient control-path design method based on the concept of checkpoint-insertion using error-propagation estimates.

In a digital microfluidic lab-on-chip, each fluidic operation has a specific error range associated with it, that is, its intrinsic error limit, which is defined as worst-case percentage offset of the actual output value from the nominal value.

For example, a dispensing operation with an error limit of 10% implies that the reservoir, in the worst case, can dispense a droplet with a volume of 1.1 or 0.9 times the normal value. In practice, the error limit can be obtained using laboratory experiments. These errors are typically related to droplet volumes and intermediate product concentrations.

Given a target bioassay protocol, we collect the error-limit information for every fluidic operation in the protocol. Using error analysis [Taylor 1982], the error limit of the output of an operation can be derived from the error limit of the input to the operation and the operation's intrinsic error limit. The error limit at the input of an operation is equal to the error limit of the output of its predecessor operation. We next present the error-propagation method employed here for fluidic operations such as dispensing, transportation, mixing, splitting and dilution.

An error can occur in the dispensing reservoir and lead to a sample or reagent droplet of abnormal volume. For the dispensing operation, if its intrinsic error limit is E_{Ds}, the error limit at the output is also E_{Ds}. The reservoir can therefore dispense a droplet with a volume of $(1 \pm E_{\mathrm{Ds}})$ times the normal value.

When a droplet is transported on the microfluidic array, volume loss may occur due to the absorption at the electrode surface. Therefore, we have to consider the error due to the transportation operation. We assume that the intrinsic error limit of the transportation operation is E_{Tran}. If the error limit at the start of the move operation is I, error analysis [Taylor 1982] shows us that the error limit at the end of the move operation is $\sqrt{I^2 + E_{\mathrm{Tran}}^2}$. The underlying assumption here is that the intrinsic errors are independent Gaussian random variables [Taylor 1982].

When two droplets D_1 and D_2 are mixed, due to volume loss during mixing (e.g., due to evaporation and absorption), the volume of the output droplet may not equal the sum of the volumes of D_1 and D_2. We assume that the intrinsic error limit of the mix operation is E_{Mix}. If the error limits for D_1 and D_2 at the start of the mix operation are I_1 and I_2, respectively, we use the error analysis method of Taylor [1982] and obtain the error limit at the end of the mix operation to be $\sqrt{(0.5I_1)^2 + (0.5I_2)^2 + E_{\mathrm{Mix}}^2}$.

During the split operation, in many cases, a larger droplet cannot be evenly split into two droplets due to a small difference in the voltages applied on the two electrodes adjacent to the droplet. We assume that the intrinsic error limit for the split operation is E_{Slt}. If the error limit for the droplet at the start of the split operation is I, we derive the error limit at the end of the split operation to be $\sqrt{I^2 + (2E_{\mathrm{Slt}})^2}$.

On-chip dilution is performed by combining a mix operation with a subsequent split operation. The mixing of a sample droplet D_{S} of concentration C and a unit buffer droplet D_{B} ideally results in a droplet with twice the unit volume, and concentration $C/2$. Splitting this large droplet results in the ideal case in two unit-volume droplets of concentration $C/2$ each. Therefore, the error estimate for dilution is the combination of the error estimates for the mix operation and the subsequent split operation. We assume that the intrinsic error limit for dilution is E_{Dlt}. If the error limits at the two input droplets are I_1 and I_2, respectively, we calculate the error limit at the output to be $\sqrt{(0.5I_1)^2 + (0.5I_2)^2 + (2E_{\mathrm{Dlt}})^2}$.

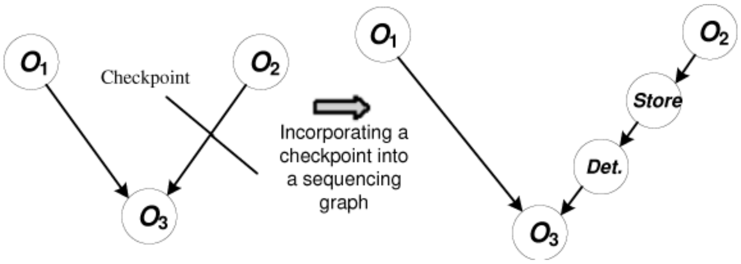

Fig. 3. The incorporation of a checkpoint in a sequencing graph (*Det.* refers to detection operation).

We apply the above error-propagation analysis to the bioassay sequencing graph and calculate the error-limit for the output of each fluidic operation. The error limit at the input of an operation is equal to the error limit at the output of its predecessor operation. The magnitude of the error limit is increased as more operations are considered in the sequencing graph. At some point, the derived output error limit will exceed a predetermined threshold $E_{\text{threshold}}$, which is obtained from the precision required for the protocol and the detector sensitivity. At this point, a checkpoint must be added. After inserting the checkpoint, the error limit for the output of this fluidic operation is set to 0. We continue to calculate the error limit for the outputs of the subsequent fluidic operations until we reach the end of the protocol. In this way, the error-propagation-based checkpoint-allocation method reduces the number of checkpoints while maintaining coverage for all the possible failures during assay operation.

5. CONTROL-PATH SYNTHESIS

Next we discuss the implementation of the control path, a step referred to as control-path synthesis. The goal is to incorporate control paths into the synthesis of a bioassay. We incorporate checkpoints using the PRSA-based unified synthesis method. A checkpoint is simply the storage of the intermediate product droplet, and a subsequent detection is used for error detection. Therefore, a storage operation and a subsequent detection operation are inserted into the sequencing graph at the same location.

For instance, a checkpoint located between operations O_2 and O_3 in Figure 3 is mapped to a storage operation and a subsequent detection operation at the same position. By applying the PRSA-based synthesis method to this modified sequencing graph, we are able to incorporate the control path as part of the bioassay protocol. The pseudocode for the control-path incorporation unified synthesis method is shown in Figure 4. The parameters for the PRSA-based synthesis method are taken from Su and Chakrabarty [2005]. The initial temperature of annealing process is set to 10000, and the temperature cooling rate is set to 0.9. The fine-tuning of these parameter values for a target bioassay is left for future work.

The 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 completion time (denoted by $T/T_{\max}$), respectively. Note that $A_{\max}$ and $T_{\max}$ refer to the maximum allowable array area and bioassay completion time, respectively. The

```
Set initial intrinsic error limits for all the fluidic operations;
Set error-limit threshold for the bioassay;
Allocate checkpoints based on error-propagation analysis;
Set initial population of chromosome and the initial temperature;
Implement the synthesis using the information of initial chromosomes: {Phase 1: Resource binding; Phase 2: Scheduling; Phase 3: Placement}
while Stopping criteria of annealing is not satisfied do
    for $i = 1$ to $N$ do
        Reproduction; Crossover:
        {Parameterized uniform crossover is employed to generate the child chromosome from two randomly-selected parent chromosomes;
        if Fitness(child) $<$ Fitness(parents) or rand(0, 1) $<$ exp($-$[Fitness(child)$-$Fitness(parents)]/T) then
            Child chromosome is selected;
        else
            Parent chromosome (the best one) is selected;
        end if}
        Mutation; New population replaces the old generation;
    end for
    T = rate $\times$ T;
end while
Find the chromosome with the best fitness value from the final population after the annealing process;
Output the results of resource binding, scheduling and placement.
```

Fig. 4. PRSA-based synthesis procedure with checkpoint insertion.

solution with the lowest value of the metric $(\alpha \times A/A_{\max} + (1-\alpha) \times T/T_{\max})$ is considered to a desirable "design point", where α can be fine-tuned during simulation and based on application requirements.

6. SOFTWARE PROGRAMS AND IMPLEMENTATION FOR ROLLBACK RECOVERY

For each fluidic operation in the sequencing graph, the bioassay synthesis results that are obtained using the synthesis method in Section 5 indicate the resource that is used to implement the operation, the start time and end time during which the operation is implemented, and the location where the operation is implemented on the microfluidic array. Note that the bioassay synthesis results also include the fluidic operations for checkpoints.

The bioassay synthesis results are mapped to a software program and stored in microcontroller memory. In the software program, each line is a command that corresponds to a specific fluidic operation in the control-path based bioassay. The first section of the command line is the address where the command is stored in the microcontroller memory; the second section is the corresponding fluidic operation; the rest of the commands are the start time and end time during which the fluidic operation is implemented, the resource that is utilized to implement the operation, and the location where it is implemented on the microfluidic array. For example, Table II shows the software program

Table II.
The software program corresponding to the bioassay synthesis results from the sequencing graph in Figure 2

Address	Fluidic Peration	Duration (clock cycle) Start—End	Resource	Module Placement (2-D coordinate)
0083	O_0	0–6	4-electrode mixer	(2,2)
0084	C_1	7–12	Detector 1	(1,1)
0085	O_1	13–21	2 × 3-array dilutor	(3,3)
0086	O_2	22–27	2 × 4-array dilutor	(2,4)
0087	C_2	28–33	Detector 1	(1,1)
0088	O_5	7–15	2 × 3-array dilutor	(5,6)
0089	C_3	16–21	Detector 2	(10,1)
0090	O_3	30–35	2 × 4-array dilutor	(6,2)
0091	O_4	36–42	4-electrode mixer	(4,6)

corresponding to synthesis results of the sequencing graph with control paths shown in Figure 2.

For checkpoints C_1, C_2 and C_3, each of their re-execution subroutines corresponds to a fragment of the program (subprogram) that can be identified by the starting address and end address in the microcontroller memory. For example, since the re-execution subroutine of checkpoint C_2 includes fluidic operations O_1 and O_2, the corresponding subprogram for checkpoint C_2 starts from Address 0085 and ends at Address 0087. Similarly, the subprogram for checkpoint C_1 starts from Address 0083 and ends at Address 0084. The subprogram for checkpoint C_3 starts from Address 0088 and ends at Address 0089.

At the beginning, the microcontroller starts to implement all the commands whose start times are 0. A counter is used to record current time. As time advances, the micro-controller exhaustively searches all the commands in the memory, and implement commands whose fluidic operations start at the current time. For each command, the microcontroller will allocate the required resource, module-placement location, and duration for the corresponding fluidic operation. Based on this process, the bioassay is implemented according to the synthesis results by interpreting the programs stored in the controller memory.

During bioassay execution, the detection output for each checkpoint is sent to the microcontroller that coordinates bioassay execution. If an error is detected at a certain checkpoint, the microcontroller intercepts the program for the bioassay and points to the starting address of the corresponding subprogram of the re-execution subroutine. The microcontroller will save current status of the bioassay and re-execute the subprogram to correct the errors. When the subroutine is finished, the microcontroller reloads the previously saved bioassay status, and continues the normal bioassay according to the synthesis results. Note that the counter stops during the re-execution process, and resumes when the normal bioassay is continued.

For example, at clock cycle 33, the detection output of checkpoint C_2 is sent to the micro-controller. If an error is detected at C_2, it implies that the error is localized among operations between C_1 and C_2. In this case, the microcontroller will implement the following steps: first, at clock cycle 33, the microcontroller stops all the concurrently implemented fluidic operations, for example, operation O_3 that starts from clock cycle 30 and ends at clock cycle 35. All the intermediate droplets of these concurrently-implemented operations are moved to on-chip storage units. The time counter stops at clock cycle 33. Second, the microcontroller points to the start address of the subprogram for checkpoint C_2 (0085) and implement the subprogram to correct errors. The subprogram includes three commands corresponding to fluidic operations O_1, O_2 and C_2. Note that input droplets of the subprogram of checkpoint C_2, that is, copy droplets of operations O_0 and O_5, will be retrieved from on-chip storage units and consumed during the re-execution. After the re-execution of the subprogram of checkpoint C_2, the errors are corrected. Next the microcontroller retrieves the intermediate droplets of the concurrently implemented operations (e.g., operation O_3), and resumes the normal bioassay according to the synthesis results. The counter continues from clock cycle 33.

7. EVALUATION FOR BIOASSAYS

In this section, we evaluate synthesis results with the incorporation of control paths into bioassays. We also evaluate the completion time of bioassays and the resource usage when the subroutines corresponding to the checkpoints are re-executed due to the detection of errors.

7.1 Protein Assay

We first evaluate the synthesis results with the incorporation of control paths into a real-life protein assay. A colorimetric protein assay has been carried out on a digital microfluidic lab-on-chip [Srinivasan et al. 2004]. Based on the Bradford reaction [Srinivasan et al. 2004], 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 lab-on-chip. Buffer droplets, such as 1M NaOH solution, are then introduced to dilute the sample to obtain a desired dilution factor (*DF*). 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 $DF = 2^N$ 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 to measure the protein concentration.

Finally, after the assay is completed, all droplets are transported from the array to the waste reservoir. Figure 5 shows a sequencing graph model for the above protocol $DF = 128$. 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

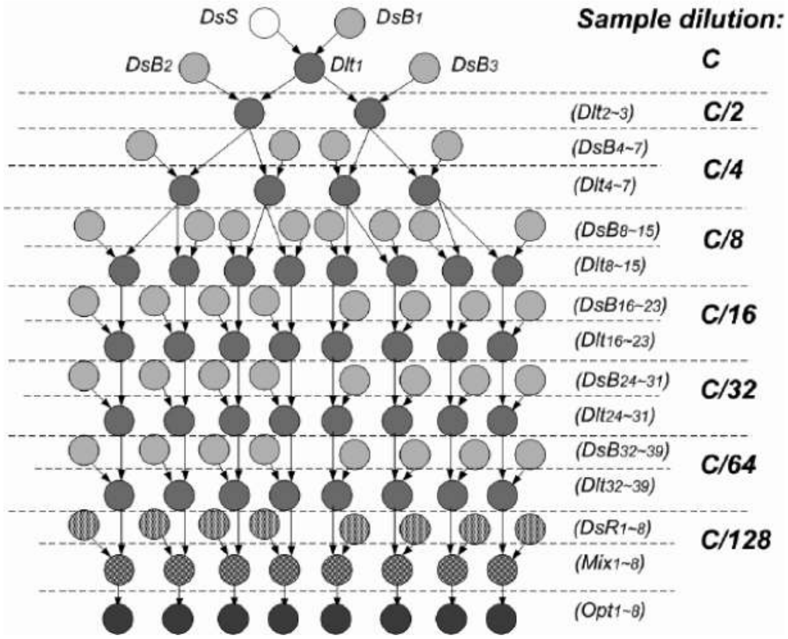

Fig. 5. Sequencing graph for a protein assay [Srinivasan et al. 2004].

($i = 1, \ldots, 39$) denotes the binary dilution (including mixing/splitting) operations, Mix_i ($i = 1, \ldots, 8$) represents the mixing of diluted sample droplets, and reagent droplets; Opt_i ($i = 1, \ldots, 8$) denotes the optical detection of the droplets. Until the fourth step of 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 its corresponding on-chip storage unit. This droplet can subsequently be retrieved during the re-execution subroutine.

We also need to specify some design parameters for the lab-on-chip to be synthesized. As an example, we set the maximum allowable completion time for the protein assay to be 500 seconds. We assume that there is only one on-chip reservoir/dispensing port available for sample fluids, but there are two such ports for buffer fluids, two for reagent fluids, and one for waste fluids. The duration for the detection operation is set to 5 seconds.

We first insert checkpoints into the sequencing graph of the protein assay. Without loss of generality, and on the basis of experimental evidence, the intrinsic error limit of the dispensing operation E_{Ds} is set to 8%, the intrinsic error limit of the transportation operation E_{Tran} is set to 12%, the intrinsic error limit of the mixing operation E_{Mix} is set to 10%, and the intrinsic error limit of the dilution E_{Dlt} is set to 8%. We consider different error-limit thresholds $E_{\mathrm{threshold}}$ for the entire protein assay. By applying error-propagation analysis to the fluidic operations of the protein assay, we obtain error limits for the outputs of fluidic operations and the corresponding checkpoints.

Table III.
Error limits (*Err*.) and checkpoints (*CP*, Yes: Y; No: N) under different values of $E_{threshold}$

Concentration		C	$C/2$	$C/4$	$C/8$	$C/16$	$C/32$	$C/64$
$E_{\text{threshold}}$ $= 30\%$	*Err.*	18.97%	22.49%	24.05%	25.15%	26.12%	27.03%	27.90%
	CP	N	N	N	N	N	N	N
$E_{\text{threshold}}$ $= 25\%$	*Err.*	18.97%	22.49%	24.05%	25.15%	22.89%	26.29%	24.41%
	CP	N	N	N	Y	N	Y	N
$E_{\text{threshold}}$ $= 23\%$	*Err.*	18.97%	22.49%	24.05%	22.09%	25.42%	23.66%	24.41%
	CP	N	N	Y	N	Y	Y	Y
$E_{\text{threshold}}$ $= 15\%$	*Err.*	18.97%	20.20%	21.26%	22.09%	22.89%	23.66%	24.41%
	CP	Y	Y	Y	Y	Y	Y	Y

Table III lists the error limits and checkpoints for different values of $E_{\text{threshold}}$. For each threshold value, Table III shows the error limits for the outputs of dilution operations *Dlt* at different sample concentrations, and whether checkpoints must be added at these outputs ("Y" or "N"). Note that the number of inserted checkpoints depends on the value of $E_{\text{threshold}}$. As shown in Table III, for $E_{\text{threshold}} = 30\%$, since the error limits for the outputs of dilution operations *Dlt* at all the sample concentrations are less than $E_{\text{threshold}}$, no checkpoint is added. For $E_{\text{threshold}} = 25\%$, the error limit for the outputs of dilution operations at sample dilution $C/8$ is larger than $E_{\text{threshold}}$. Therefore, checkpoints are added at the outputs of these dilution operations, and the error limits of the outputs of these dilution operations are set to zero for the error-limit calculation of the subsequent dilution operations. We continue the error-limit calculation until we reach the leaf nodes of the sequencing graph for the protein assay. Since the error limits for the outputs of dilution operations at sample dilution $C/32$ are also larger than $E_{\text{threshold}}$, checkpoints are added here. Altogether, 16 checkpoints are added when $E_{\text{threshold}} = 25\%$. Note that when $E_{\text{threshold}}$ decreases, more checkpoints are added. For example, when $E_{\text{threshold}} = 15\%$, checkpoints are added for the dilution operations at all the concentrations from C to $C/64$, that is, there are 39 checkpoints in total.

Next we incorporate checkpoints into the protein assay. We utilize the PRSA-based unified synthesis method to derive chip designs based on these modified sequencing graphs with different checkpoint sets for various values of $E_{\text{threshold}}$. Figure 6 lists the assay completion times for control-path-based protein assay for different error-limit thresholds for a 10×10 microfluidic array. We consider two cases: four detectors and three detectors for normal bioassay and checkpointing (storage and error detection). Here we assume that no error occurs during the assay execution. Therefore, the assay completion times reported here include the time needed for normal bioassay operations and checkpointing. Note that no checkpoint is added under $E_{\text{threshold}} = 30\%$, therefore the assay completion time for this case is equal to that without control paths. For both 4-detector case and 3-detector case, the results show that when $E_{\text{threshold}}$ becomes lower, more checkpoints are incorporated into the protein assay, thereby the time for checkpointing increases, and the assay completion time also increases. The percentage increases over the baseline of no checkpoint are shown in the figure. For the same number of checkpoints, the assay completion time for the

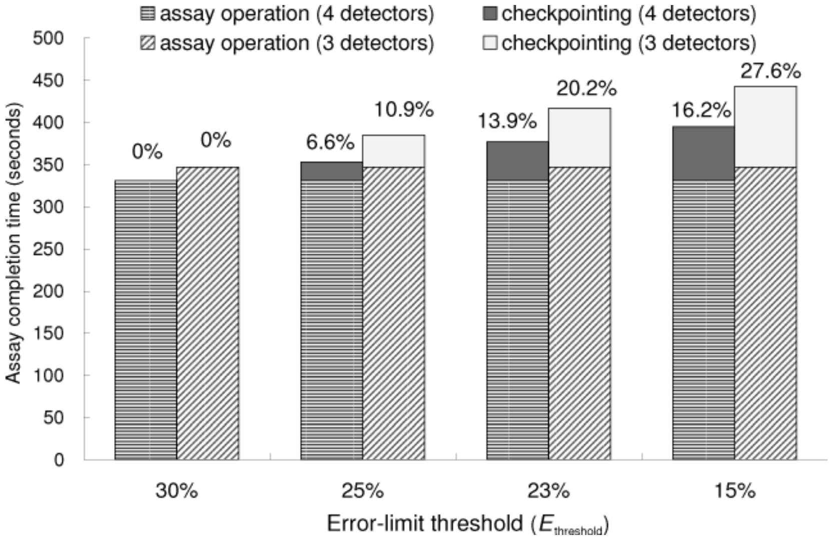

Fig. 6. Assay completion times of using 4 detectors and 3 detectors for normal bioassay and checkpointing for various error-limit thresholds and increase in assay time compared to no control path. The percentages on the bars refer to the increase in the assay time (compared to the assay without control flow) when there are no errors.

4-detector case is lower than that for the 3-detector case, since more detection operations can be executed concurrently in the 4-detector case. The experiment was performed on a 2.0 GHz Intel Core2 Dual processor, with 1 GB of memory. The CPU time needed was 35 m for $E_{\text{threshold}} = 15\%$.

Next we evaluate the effect of microfluidic array size on the assay completion times. Figure 7 lists the assay completion times for control-path-based protein assay for different error-limit thresholds in 10×10 and 15×15 microfluidic arrays. Four detectors are used here for both normal bioassay and checkpointing. Here we assume that no error occurs during the assay execution. For both 10×10 and 15×15 microfluidic arrays, the results show that when $E_{\text{threshold}}$ becomes lower, more checkpoints are incorporated into the protein assay, thereby the assay completion time increases. The percentage increases over the baseline of no checkpoint are shown in the figure. For the same checkpoints, the assay completion time for a 15×15 microfluidic array is lower than that for a 10×10 microfluidic array, since more fluidic modules such as mixers and diluters can be simultaneously placed on the larger array.

The synthesis results of the control-path-based protein assay are mapped to a software program and stored in microcontroller memory. Table IV shows a fragment of the program corresponding to synthesis results of sample concentration $C/4$ in the sequencing graph of protein assay in Figure 5. C_4 to C_7 are the checkpoints for operations Dlt_4 to Dlt_7. Here we integrate the control paths including checkpoints and re-execution subroutines for $E_{\text{threshold}} = 15\%$ in the bioassay. For example, at clock cycle 67, the detection output of checkpoint C_6 is sent to the microcontroller. If an error is detected at C_6, the microcontroller

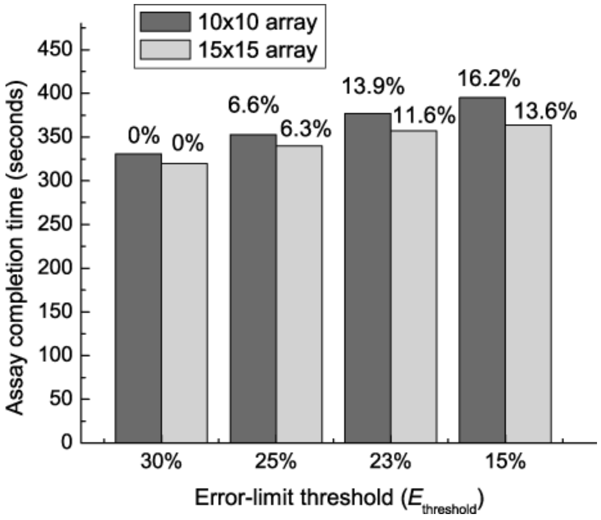

Fig. 7. Assay completion times for various error-limit thresholds in 10 × 10 and 15 × 15 microfluidic arrays and increase in assay time compared to no control path. The percentages on the bars refer to the increase in the assay time (compared to the assay without control flow) when there are no errors.

Table IV.
The software program corresponding to the bioassay synthesis results of sample concentration $C/4$ in Figure 5

Address	Fluidic Operation	Duration (clock cycle) Start—End	Resource	Module Placement (2-D coordinate)
0011	Dlt_4	46–53	4-electrode dilutor	(3,1)
0012	C_4	54–59	Detector 1	(1,1)
0013	Dlt_5	76–81	2 × 4-array dilutor	(5,3)
0014	C_5	82–87	Detector 3	(5,1)
0015	Dlt_6	56–61	2 × 4-array dilutor	(1,5)
0016	C_6	62–67	Detector 1	(1,1)
0017	Dlt_7	58–70	2 × 2-array dilutor	(5,3)
0018	C_7	71–76	Detector 2	(1,10)

will implement the corresponding subprogram (from Address 0015 to Address 0016) for the re-execution subroutine of C_6, that is, operation Dlt_6, to correct the errors.

Next we evaluate the assay completion time if errors occur during assay execution. We consider errors that are detected for different sample concentrations. Figure 8 compares the assay completion time with control paths to that without control paths when an error is detected at the concentration levels indicated on the x-axis. We integrate the control paths including checkpoints and re-execution subroutines for $E_{\text{threshold}} = 15\%$ in the bioassay. The assay completion times reported here include the time needed for normal bioassay operations, checkpointing (storage and error detection), and rollback recovery. The results show that if errors occur at any specific sample concentration level, it takes more time for the lab-on-chip without control paths to complete the

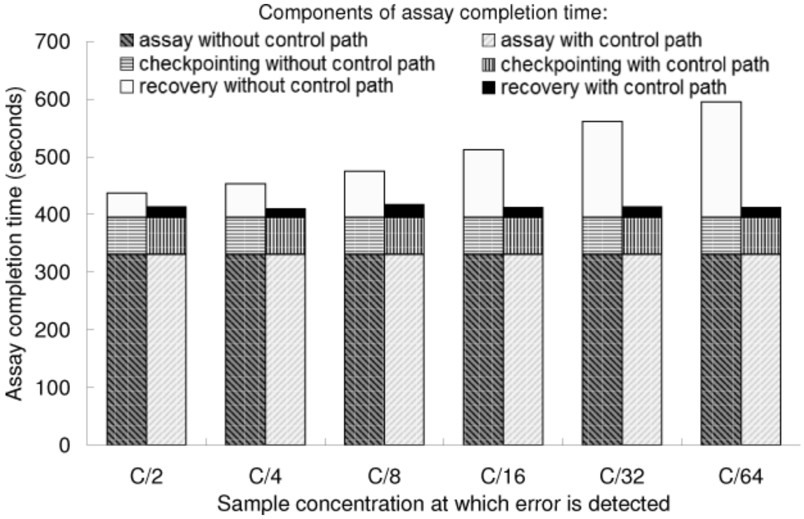

Fig. 8. Assay completion time comparison for the scheduled protein assay protocol, with and without control paths, when errors appear at intermediate points (sample concentrations).

assay than that with control paths. For example, if errors occur at sample concentration $C/32$, for the assay protocol without control paths, the entire bioassay must be repeated. Therefore, it takes 561 seconds to complete the assay. However, if we incorporate control paths into the assay protocol, we only need to implement the re-execution subroutines corresponding to checkpoints at the outputs of these dilution operations, that is, implement all fluidic operations between sample concentration $C/16$ and $C/32$. Therefore, it takes only 413 seconds to complete the assay. When errors occur at a relatively high concentration level, for example, at $C/4$, the protocol without control paths returns to the start of bioassay, and the protocol with control paths returns to checkpoints at $C/2$. There is only one level of dilution between the start and $C/2$. Therefore, there is only slight difference between the assay completion times. The difference is more striking when errors occur at the dilution operations at lower sample concentration, for example, at $C/32$, since the difference between the completion time from the start and from the immediate upstream checkpoints becomes larger.

Next we evaluate the resource usage if errors occur during assay execution. Here, resource usage refers to the consumption of droplets (sample, reagent and buffer) during the bioassay. Figure 9 compares the number of droplets consumed for the protein assay with control paths to that without control paths when errors are detected at the concentration levels indicated on the x-axis. If errors occur at any specific concentration level, the protein assay without control paths consumes more droplets than that with control paths. For example, if errors occur at the dilution operations at sample concentration $C/32$, for the protein assay without control paths, the entire bioassay must be repeated, and 80 droplets are consumed to complete the bioassay. However, the protein assay with control paths only needs to re-execute the corresponding

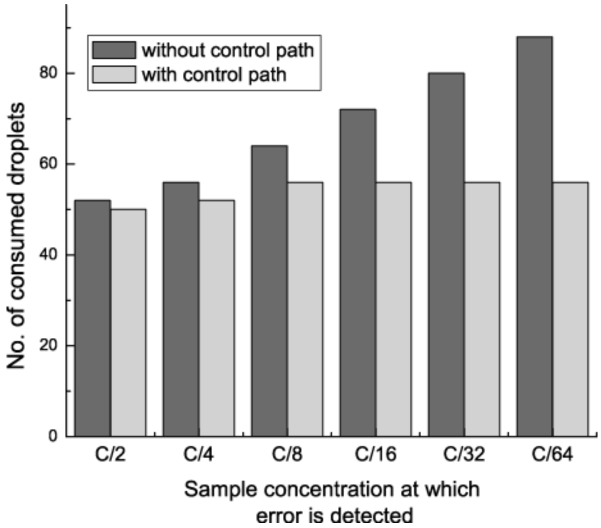

Fig. 9. Droplet consumption comparison for the scheduled protein assay protocol, with and without control paths, when errors appear at intermediate points (sample concentrations).

subroutine to correct the error, that is, implement all fluidic operations between $C/16$ and $C/32$, and only 56 droplets are consumed overall.

The error-recovery capability of the control-path design for different numbers of checkpoints can be measured using the bioassay completion time when errors occur. When a certain number of errors are randomly inserted into the bioassay protocol with varying number of checkpoints, the control-path design that achieves less completion time has higher error-recovery capability. Next we inject multiple errors at randomly chosen concentration levels of the protein assay, in order to evaluate the error-recovery capability of the control-path design with different numbers of checkpoints. Each randomly chosen dilution level is associated with a probability of an error. The assay completion time and number of consumed droplets with control paths is averaged over 100 runs. For each error-limit threshold $E_{\text{threshold}}$, we obtain the corresponding checkpoint set and the scheduled protein assay protocol with control paths. We calculate the average assay completion time and number of consumed droplets for different values of $E_{\text{threshold}}$. Since there are 39 dilution operations altogether, at most 39 errors can be injected.

Figure 10 and Figure 11 shows the average assay completion times and average number of consumed droplets, respectively, for various error-limit thresholds, when multiple errors are injected at randomly chosen concentration levels. For $E_{\text{threshold}} = 15\%, 23\%$, or 25%, when the number of injected errors increases, the average assay completion time and the average number of consumed droplets increase, since more rollback is necessary to correct the errors. For a specific number of injected errors, the average assay completion time and the average number of consumed droplets for low error-limit threshold (e.g., $E_{\text{threshold}} = 15\%$) is less than that for high error-limit threshold (e.g.,

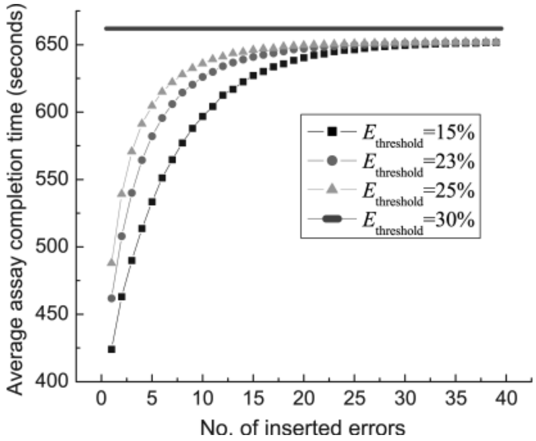

Fig. 10. Average assay completion time for the scheduled protein assay protocol for various error-limit thresholds, when different numbers of errors are injected at randomly chosen dilutions.

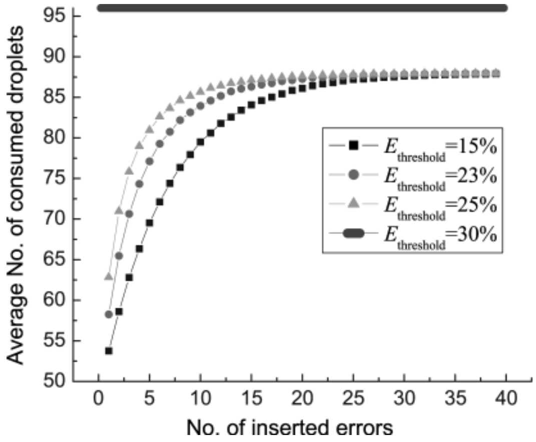

Fig. 11. Droplet consumption for the scheduled protein assay protocol for various error-limit thresholds, when different numbers of errors are injected at randomly chosen dilutions.

$E_{\text{threshold}} = 25\%$). This is because to correct a specific group of errors, the re-execution subroutines of low value of $E_{\text{threshold}}$ contain less fluidic operations, while the re-execution subroutines of large value of $E_{\text{threshold}}$ contain more fluidic operations. The horizontal bar ($E_{\text{threshold}} = 30\%$) corresponds to the case when no checkpoints are inserted. In this case, the assay completion time and the number of consumed droplets are independent of the number of errors, since the entire assay must be re-executed.

The average completion time and the average number of consumed droplets for different $E_{\text{threshold}}$ trends to saturate as we increase the number of injected errors. This is because there are altogether eight concentration levels in the protein assay, and a subroutine is associated with each concentration level. This subroutine is executed to correct all the errors at the concentration level. When a large number of errors are injected, these errors trend to be distributed among the different concentration levels, but at most eight subroutines are available to correct all the errors. The maximum average assay completion time is 652 clock cycles, and the maximum average number of consumed droplets is 88, when all the eight subroutines are implemented during the bioassay with control paths. For the baseline case with no control path, the assay completion time is 662 clock cycles, and the number of consumed droplets is 96.

7.2 Interpolating Mixing Architecture

We next evaluate the synthesis results with the incorporation of control paths into an interpolating mixing architecture [Ren et al. 2003]. Not only an exponential dilution of 2^N can be obtained using a recursive manner of binary dilutions, but also the two-fold dilution step can be extended to two droplets of different concentrations C_1 and C_2, which would result in two unit droplets with an interpolated concentration of $(C_1 + C_2)/2$ each. By cascading the exponential and interpolating dilution steps in a serial fashion, arbitrary dilution factors can be obtained. For example, by mixing and splitting two unit droplets of concentration $C/8$ and $C/16$, we can obtain a concentration $C/10.67$. This scheme of obtaining the desired dilution ratio is referred to as interpolating serial dilution.

In the interpolating mixing architecture, a 0.1M KCl solution with 0.01% Triton-X and colored with a red food dye is used as the sample liquid. A 0.1M KCl solution with 0.01% Triton-X is used as the dilution buffer. Optical detectors (e.g., a LED-photodiode setup) are used to measure the concentration of the droplets. Figure 12 shows a sequencing graph model for this protocol to obtain four different concentrations $C/10.67$, $C/21.33$, $C/42.67$, and $C/85.33$.

Since the complexity lies in the fact that the given architecture requires accurate dilution results, volume error should be minimized in order to maintain the concentration accuracy. We incorporate checkpoints into the interpolating mixing architecture and utilize the PRSA-based unified synthesis method to derive chip designs based on these modified sequencing graphs with different checkpoint sets for various values of $E_{\text{threshold}}$. Figure 13 lists the assay completion times for control-path-based interpolating mixing architecture for different error-limit thresholds for a 10 × 10 microfluidic array. Four detectors are used for normal bioassay and checkpointing (storage and error detection). Since we assume that no error occurs during the assay execution, the assay completion times reported here include the time needed for normal bioassay operations and checkpointing. Note that no checkpoint is added under $E_{\text{threshold}} = 30\%$, thereby the assay completion time for this case is equal to that without control paths. Under $E_{\text{threshold}} = 18\%$, checkpoints are added after all the dilution operations. The results show that for lower value of $E_{\text{threshold}}$, more checkpoints are incorporated into the interpolating mixing architecture, thereby the assay completion time increases.

Figure 14 compares the assay completion time with control paths to that without control paths when an error is detected at the concentration levels

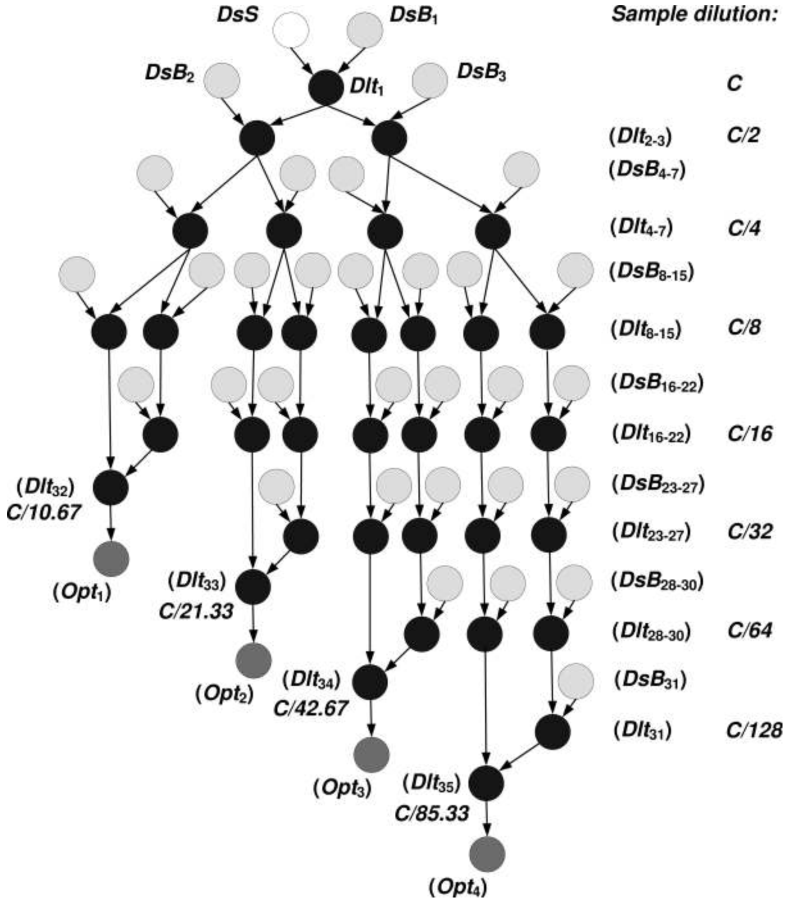

Fig. 12. Sequencing graph for an interpolating mixing architecture [Ren et al. 2003].

indicated on the x-axis. We integrate the control paths including checkpoints and re-execution subroutines for $E_{\text{threshold}} = 18\%$ in the bioassay. The assay completion times reported here include the time needed for normal bioassay operations, checkpointing (storage and error detection), and rollback recovery. The results show that if errors occur at any specific sample concentration level, more time will be spent for the lab-on-chip without control paths to complete the assay than that with control paths. Note that without control paths, when errors are detected at $(C/16 + C/32)/2$, that is, $C/21.33$, more assay time are needed to recover from the errors than the cases for $C/16$ and $C/32$, since the dilution operation for $C/21.33$ is the successor of both $C/16$ and $C/32$. Similar results are observed for $C/10.67$, $C/42.67$ and $C/85.33$.

Next we inject multiple errors at randomly chosen concentration levels of the interpolating mixing architecture, in order to evaluate the error-recovery capability of the control-path design with different numbers of checkpoints. Figure 15 shows the average assay completion times for various error-limit

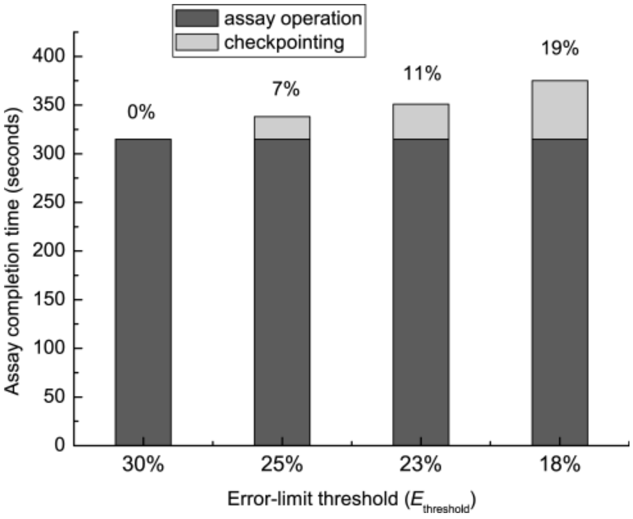

Fig. 13. Assay completion times for normal bioassay and checkpointing for various error-limit thresholds and increase in assay time compared to no control path. The percentages on the bars refer to the increase in the assay time (compared to the assay without control flow) when there are no errors.

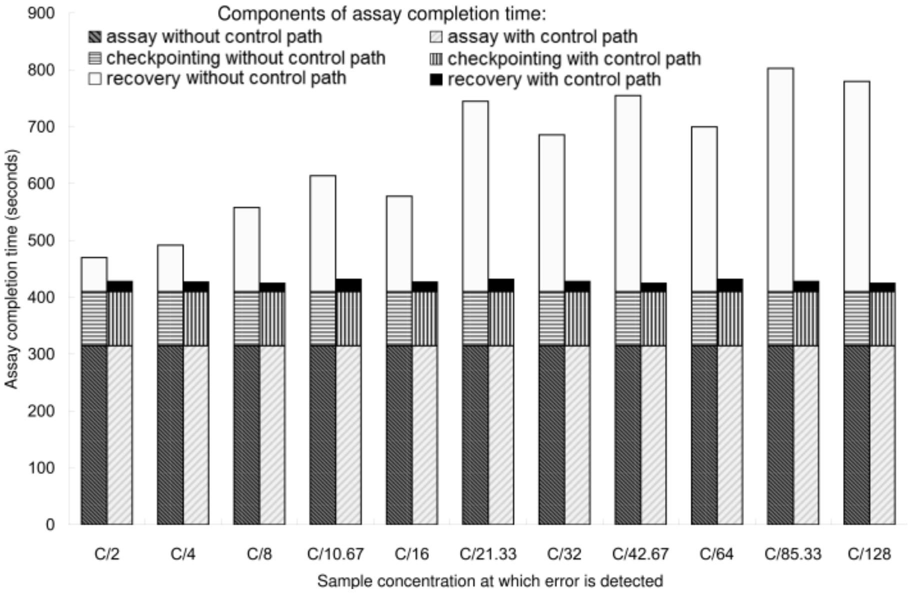

Fig. 14. Assay completion time comparison for the scheduled interpolating mixing architecture, with and without control paths, when errors appear at intermediate points (sample concentrations).

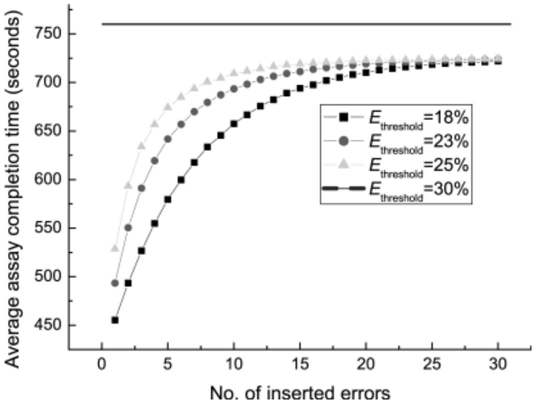

Fig. 15. Average assay completion time for the scheduled interpolating mixing architecture for various error-limit thresholds, when different numbers of errors are injected at randomly chosen dilutions.

thresholds, when multiple errors are injected at randomly chosen concentration levels. For $E_{\text{threshold}} = 18\%, 23\%$, or 25%, when the number of injected errors increases, the average assay completion time increases, since more rollback is necessary to correct the errors. For a specific number of injected errors, the average assay completion time for a low error-limit threshold (e.g., $E_{\text{threshold}} = 18\%$) is less than that for a high error-limit threshold (e.g., $E_{\text{threshold}} = 25\%$). The horizontal bar ($E_{\text{threshold}} = 30\%$) corresponds to the case when no checkpoints are inserted. The average completion time for different $E_{\text{threshold}}$ trends to saturate as we increase the number of injected errors.

We also incorporate control paths into the mixing stage of the polymerase chain reaction (PCR), as well as randomly generated sequences of fluidic operations (synthetic benchmarks). Table V shows the error-recovery capability and the corresponding time cost of control paths for different values of $E_{\text{threshold}}$ for each bioassay. The error-recovery capability is evaluated using the bioassay completion time with control-path design when a constant number of errors (e.g., 5 errors) are injected at randomly chosen locations of the bioassay protocol. The time cost of control paths includes the time needed for checkpointing (storage and error detection) and rollback recovery (i.e., the re-execution of the subroutine associated with checkpoints).

In Table V, for each bioassay, when error-limit threshold $E_{\text{threshold}}$ decreases, more checkpoints are inserted for the control-path design, the assay completion time with the control-path design for randomly-injected errors decreases, and the corresponding time cost of control paths also decreases. This is because to correct these errors, the re-execution subroutines for low $E_{\text{threshold}}$ contain less fluidic operations, while the re-execution subroutines for high $E_{\text{threshold}}$ contain more fluidic operations.

8. CONCLUSIONS

We have presented a unified synthesis method that incorporates control paths in the design of a digital microfluidic lab-on-chip for bioassay protocols. We have

Table V.
Comparison of error-recovery capability and time cost of control paths for different values of $E_{\text{threshold}}$

Bioassay	$E_{\text{threshold}}$	No. Checkpoint	Biossay Completion Time (seconds)	Time Cost of Control Paths (seconds)
Protein assay	30%	0	642	311
	25%	16	605	274
	23%	28	582	251
	15%	39	533	202
Interpolating mixing architecture	30%	0	760	405
	25%	13	674	319
	23%	20	642	287
	18%	31	579	224
PCR	30%	0	56	26
	20%	3	51	21
	15%	7	48	18
Synthetic 1	30%	0	610	307
	25%	18	575	272
	15%	31	503	201
Synthetic 2	30%	0	876	423
	25%	22	662	209
	20%	39	629	176
	15%	55	550	109

proposed an efficient control-path design method based on error-propagation estimates for different fluidic operations. The proposed synthesis method allows the incorporation of control paths and synthesis of the bioassays to be carried out simultaneously. We have demonstrated the advantages of this approach using real-life bioassay applications. We have shown that if errors occur during bioassay execution, it takes considerably less time and consumes less droplets for the assay protocol with control paths to complete the assay than the assay protocol without control paths.

REFERENCES

Bohringer, K. F. 2006. Modeling and controlling parallel tasks in droplet-based microfluidic systems. *IEEE Trans. Comput.- Aid. Des. Integr. Circ. Syst. 25*, 334–344.

Chakrabarty, K. and Su, F. 2006. *Digital Microfluidic Biochips: Synthesis, Testing, and Reconfiguration Techniques*. CRC Press, Boca Raton, FL.

Chakrabarty, K. and Zeng, J. 2005. Design automation for microfluidics-based biochips. *ACM J. Emer. Technol. Comput. Syst. 1*, 186–223.

Chakrabarty, K. and Zeng, J. 2006. *Design Automation Methods and Tools for Microfluidics-Based Biochips*. Springer, Berlin.

Cho, M. and Pan, D. Z. 2008. A high-performance droplet routing algorithm for digital microfluidic biochips. *IEEE Trans. Comput.- Aid. Des. Integr. Circ. Syst. 27*, 1714–1724.

Cho, S.-Y., Seo, S.-W., Brooke, M. A., and Jokerst, N. M. 2002. Integrated detectors for embedded optical interconnections on electrical boards, modules, and integrated circuits. *IEEE J. Select. Topics Quantum Electron. 8*, 1427–1434.

Fair, R. B., Khlystov, A., Tailor, T. D., Ivanov, V., Evans, R. D., Griffin, P. B., Srinivasan, V., Pamula, V. K., Pollack, M. G., and Zhou, J. 2007. Chemical and biological applications of digital-microfluidic devices. *IEEE Des. Test Comput. 24*, 10–24.

GRIFFITH, E. J., AKELLA, S., AND GOLDBERG, M. K. 2006. Performance characterization of a reconfigurable planar-array digital microfluidic system. *IEEE Trans. Comput.- Aid. Des. Integr. Circ. Syst. 25*, 345–357.

LUAN, L., EVANS, R. D., JOKERST, N. M., AND FAIR, R. B. 2008. Integrated optical sensor in a digital microfluidic platform. *IEEE Sensors J. 8*, 628–635.

MADSEN, B. C. AND MURPHY, R. J. 1981. Flow-injection and photometric determination of sulfate rainwater with methythymol blue. *Analyt. Chem. 53*, 1924–1926.

MAFTEI, E., POP, P., MADSEN, J., AND STIDSEN, T. 2008. Placement-aware architectural synthesis of digital microfluidic biochips using ilp. In *Proceedings of the International Conference on Very Large Scale Integration of System on Chip*. 425–430.

MAHFOUD, S. W. AND GOLDBERG, D. E. 1995. Parallel recombinative simulated annealing: a genetic algorithm. *Parall. Comput. 21*, 1–28.

MEDORO, G. 2007. Dielectrophoretic separation of human spermatozoa from epithelial cells. In *Proceedings of the International Conference on MicroTAS*.

MINAS, G., RIBEIRO, J. C., WOLFFENBUTTEL, R. F., AND CORREIA, J. H. 2005. On-chip integrated CMOS optical detection microsystem for spectrophotometric analyses in biological microfluidic systems. In *Proceedings of the IEEE International Symposium on Industrial Electronics*. 1133–1138.

POLLACK, M. G. 2001. Electrowetting-based microactuation of droplets for digital microfluidics. Ph.D. thesis, Duke University, Durham, NC.

REN, H., SRINIVASAN, V., AND FAIR, R. B. 2003. Design and testing of an interpolating mixing architecture for electrowetting-based droplet-on-chip chemical dilution. *Transduc. 1*, 619–622.

RICKETTS, A. J., IRICK, K., VIJAYKRISHNAN, N., AND IRWIN, M. J. 2006. Priority scheduling in digital microfluidics-based biochips. In *Proceedings of the Design Automation and Test in Europe (DATE)*. 329–334.

SEO, S., LEE, K. K., KANG, S., HUANG, S., DOOLITTLE, W. A., JOKERST, N. M., BROWN, A. S., AND BROOKE, M. A. 2002. The heterogeneous integration of gan thin-film metal-semiconductor-metal photodetectors onto silicon. *IEEE Photon. Technol. Lett. 14*, 185–187.

SRINIVASAN, V., PAMULA, V. K., PAIK, P., AND FAIR, R. B. 2004. Protein stamping for maldi mass spectrometry using an electrowetting-based microfluidic platform. *Proc. SPIE 5591*. 26–32.

SRINIVASAN, V., PAMULA, V. K., POLLACK, M. G., AND FAIR, R. B. 2003. Clinical diagnositics on human whole blood, plasma, serum, urine, saliva, sweat, and tears on a digital microfluidic platform. In *Proceedings of the International Conference on MicroTAS*. 1287–1290.

SU, F. AND CHAKRABARTY, K. 2005. Unified high-level synthesis and module placement for defect-tolerant microfluidic biochips. In *Proceedings of the Design Automation Conference (DAC)*. 825–830.

SU, F. AND CHAKRABARTY, K. 2008. High-level synthesis of digital microfluidic biochip. *ACM J. Emerg. Tech. Comput. Syst. 3*, 16.1–16.32.

SU, F., HWANG, W., AND CHAKRABARTY, K. 2006. Droplet routing in the synthesis of digital microfluidic biochips. In *Proceedings of the Conference and Exhibition on Design Automation and Test in Europe (DATE)*. 323–328.

TAYLOR, J. R. 1982. *An Introduction to Error Analysis: the Study of Uncertainties of Physical Measurements*. University Science Books.

XU, T. AND CHAKRABARTY, K. 2007. Functional testing of digital microfluidic biochips. In *Proceedings of the International Test Conference (ITC)*.

XU, T. AND CHAKRABARTY, K. 2008. Integrated droplet routing and defect tolerance in the synthesis of digital microfluidic biochips. *ACM J. Emerg. Technol. Comput. Syst. 4*, 3, 11.1–11.24.

XU, T., CHAKRABARTY, K., AND PAMULA, V. K. 2008. Design and optimization of a digital microfluidic biochip for protein crystallization. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD)*. 297–301.

YUH, P.-H., YANG, C.-L., AND CHANG, Y.-W. 2007a. Bioroute: A network flow based routing algorithm for digital microfluidic biochips. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD)*. 752–757.

YUH, P.-H., YANG, C.-L., AND CHANG, Y.-W. 2007b. Placement of defect-tolerant digital microfluidic biochips using the T-tree formulation. *ACM J. Emerg. Technol. Comput. Syst. 3*, 13.1–13.32.

Received August 2009; revised March 2010; accepted March 2010