# A Cyberphysical Synthesis Approach for Error Recovery in Digital Microfluidic Biochips*

Yan Luo†, Krishnendu Chakrabarty†, and Tsung-Yi Ho‡

†Electrical & Computer Engineering Dept.
Duke University, Durham, NC 27708
E-mail: {yan.luo, krish}@ee.duke.edu

‡Computer Science and Information Engineering Dept.
National Cheng Kung University, Tainan, Taiwan
E-mail: tyho@csie.ncku.edu.tw

*Abstract*—**Droplet-based "digital" microfluidics technology has now come of age and software-controlled biochips for healthcare applications are starting to emerge. However, today's digital microfluidic biochips suffer from the drawback that there is no feedback to the control software from the underlying hardware platform. Due to the lack of precision inherent in biochemical experiments, errors are likely during droplet manipulation, but error recovery based on the repetition of experiments leads to wastage of expensive reagents and hard-to-prepare samples. By exploiting recent advances in the integration of optical detectors (sensors) in a digital microfluidics biochip, we present a "physical-aware" system reconfiguration technique that uses sensor data at checkpoints to dynamically reconfigure the biochip. A re-synthesis technique is used to recompute electrode-actuation sequences, thereby deriving new schedules, module placement, and droplet routing pathways, with minimum impact on the time-to-response.**

## I. Introduction

Microfluidic biochips have now come of age, with applications to biomolecular recognition for high-throughput DNA sequencing, immunoassays, and point-of-care clinical diagnostics [1], [2]. In particular, digital microfluidic biochips, which use electrowetting-on-dielectric to manipulate discrete droplets (or "packets of biochemical payload") of picoliter volumes under clock control, are especially promising [2], [3].

The ease of reconfigurabilty and software-based control in digital microfluidics has motivated research on various aspects of automated chip design and chip application. A number of techniques have been published for architectural-level synthesis [4], module placement, and droplet routing [5], [6], [7]. However, these techniques ignore domain-specific constraints or practical realities that arise from attempting to carry out biochemical reactions and microfluidic operations on an electronic chip. Due to the randomness and complex component interactions that are ubiquitous in biological/chemical processes, predictive modeling and accuracy control are difficult [8], [9]. Yet, despite such inherent variability, many biomedical applications such as drug development and clinical diagnostics require high precision for each operation and correctness of the final result under various conditions. If an unexpected error occurs during the experiment, the outcome of the entire experiment will be incorrect. As a result, all the steps in the experiment must be repeated in order to correct the error [10].

The repetitive execution of on-chip laboratory experiments leads to the following problems: (i) wastage of samples that are difficult to obtain or prepare, and the wastage of expensive reagents; (ii) an increase in the time-to-result for a bioassay, which is detrimental to real-time detection and rapid response. Therefore, it is necessary to develop techniques for monitoring assay outcomes at intermediate stages and design an efficient error-recovery mechanism.

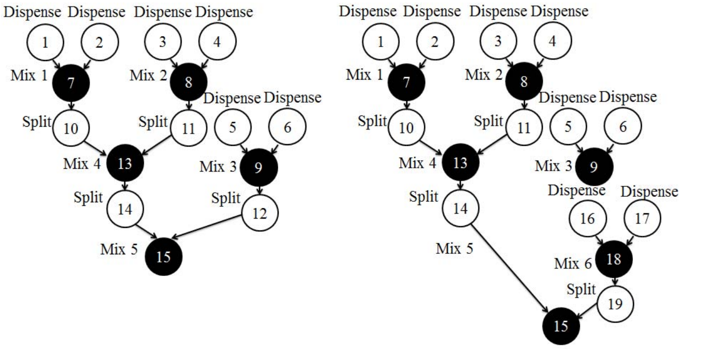
Error recovery in digital microfluidics has received relatively little attention in the literature. The only reported work is [11], which proposed intermediate stage monitoring and rollback error-recovery for a microfluidic biochip. The key idea in this work is to use optical sensors to verify the correctness of immediate product droplets at various steps in the on-chip experiment. Optical detection has been integrated with digital microfluidics to evaluate the concentration and volume of product droplets [3], [12]–[14]. In the recent approach described in [11], error recovery is carried out as follows. During bioassay execution, intermediate product droplets are sent to optical sensors. When an error is detected at an optical sensor, i.e., the volume or concentration of the droplet is below or above the acceptable calibrated range, the corresponding droplet is discarded. The operations whose outputs fail to meet the quality requirements based on sensor calibration are re-executed to generate a new product droplet to replace the unqualified droplet.

Figure 1 shows an example of rollback error-recovery. The initial sequencing graph of a bioassay is shown in Figure 1(a). Here we assume that the outputs of each dispensing, mixing and splitting operation are evaluated by an optical sensor. When an error occurs at operation 9, the system will re-execute the corresponding dispensing and mixing operations. The new sequencing graph for error recovery is shown in Figure 1(b). Operations 16, 17, 18 and 19 are added for error recovery.

In the absence of "physical aware" control software, the error recovery method in [11] suffers from three important drawbacks. The first drawback is the over-simplification of fault detection and the associated assumptions. Using a uniform "expected value" for the calibration of each optical detection operation is not practical. Note that at various stages during bioassay execution,

[*] The work of Y. Luo and K. Chakrabarty was supported in part by the US National Science Foundation under grants CCF-0914895 and CNS-1135853. The work of T.-Y. Ho was supported in part by the Taiwan National Science Council under grant NSC 100-2220-E-006-016.

978-3-9810801-8-6/DATE12/©2012 EDAA



(a)

(b)

Fig. 1. (a) Initial sequencing graph; (b) operations 16, 17, 18 and 19 are added for error recovery.

the concentration of intermediate product droplets vary in a dynamic manner, hence the calibration also needs to be repeated and carried out dynamically.

The second drawback of [11] is that all recovery operations are carried out in a stand-alone manner. When an error is detected, all other ongoing bioassay-related fluidic operations are interrupted. The potential long wait times introduced by recovery operations can lead to sample degradation and erroneous assay outcomes [15]. Some operations, such as colorimetric enzyme-kinetic reactions, require precise durations as specified by the reaction protocol, and they cannot be extended without introducing unpredictability in the experiment outcome [16].

The third drawback of [11] is that it cannot handle situations when multiple errors occur during a bioassay. For example, it does not consider the likelihood that errors can occur during recovery.

To overcome the above drawbacks, we take a transformative “cyberphysical” approach towards closed-loop and sensor feedback-driven biochip operation under program control. By exploiting recent advances in the integration of optical detectors (sensors) in a digital microfluidics biochip [13], we present a “physical-aware” system reconfiguration technique that uses sensor data at checkpoints to dynamically reconfigure the biochip.

The rest of the paper is organized as follows. Data/control transfer between the microfluidic chip and control software is described in Section II. The proposed cyberphysical re-synthesis approach is presented in Section V. Section IV shows results on assay completion times and sample/reagent volume required under various practical scenarios for three representative bioassays. Section V concludes the paper.

## II. INTERFACES BETWEEN BIOCHIP AND CONTROL SOFTWARE

The digital microfluidic system consists of the microfluidic biochip and control software. With integrated optical detectors on chip, “physical-aware” control software becomes feasible. By “physical-aware”, we mean that the software can receive information about the outcome (error-free/erroneous) of fluid-handling operations based on feedback from the optical detectors. Depending on sensor feedback, the control software can appropriately reconfigure the microfluidic biochip. In this way, the various steps in the bioassay are executed based upon real-time sensing of intermediate results.

To derive digital signals from the analog voltage outputs of optical detectors, the outputs of detectors are connected to an analog to digital converter (ADC). To transfer digitally coded values to a computer that runs re-synthesis software and stores control software, the outputs of ADC are connected to the serial port of the computer by an RS-485 interface. The control software is designed to read and operate on the data sent to the serial port. Thus, the output voltages of optical detectors are sent as feedback signals to the software.

As described in [4], biochip synthesis includes resource binding and scheduling, which specifies the start and stop times of fluidic operations. The synthesis results (control software) need to be mapped to a sequence of electrode actuation vectors consisting of “0”, “1”, and “F” (floating). A programmable logic controller (PLC) is used as the interface between the output of the control software (“controller”) and the control pins of the biochip. The controller transfers data to the memory of the PLC through the USB port of the computer and an RS-232 interface.

During bioassay execution, the PLC reads the electrode actuation sequences stored in memory and applies the appropriate sequence of voltages to the output pins. When the PLC generates output voltages based on sequences in its memory, corresponding activation voltages are applied to the electrodes of the biochip. The quality of an intermediate product in a digital microfluidic biochip can be determined by examining the product concentration in droplets through fluorescence [17], [18]. When a fluorophore tag is attached to a droplet, different product concentrations lead to emitted light of different wavelengths (i.e., different colors). This difference in color can be detected by optical sensors that convert the received light into electrical current or voltage [3]. In recent work, integrated photodetectors on the microfluidic array have been introduced [12]–[14]. Thus recent advances in the integration of a multiplicity of such miniaturized sensors provides an important motivation for the cyberphysical hardware/software co-design approach studied in this paper.

## III. “PHYSICAL-AWARE” CONTROL SOFTWARE FOR RE-SYNTHESIS

With the availability of hardware that can send feedback to the control software, it is now necessary to design physical-aware software that can analyze sensor data and dynamically adapt to it. Adaptations include updates for the schedule of fluidic handling operations, resource binding, module placement, and droplet routing pathways.

### A. Data Preparation before Bioassay Execution

The first step in data preparation is to convert the sequencing graph of the bioassay to a directed acyclic graph (DAG) and store it in memory for use by the control software. In this DAG, the vertices represent microfluidic handling operations and the edges represent precedence relations between operations. The predecessors and successors of any operation can be determined by a depth-first search on the graph [19].

The second step in data preparation is to assign error thresholds for each operation. These thresholds are determined by the requirement of precision for the bioassay and they are stored as a table in memory for use by the control software. During bioassay execution, if the optical detection result is outside the range of pre-assigned threshold values, we conclude that an error has occurred at the corresponding operation.

The last step in data preparation is computing the initial synthesis result for the bioassay. Previously published CAD methods for digital mircofluidic biochip can generate synthesis results (scheduling, binding, placement, routing) for a given biochemical protocol and a digital microfluidic platform, as well as the mapping of the synthesis results to electrode actuation sequences [4].

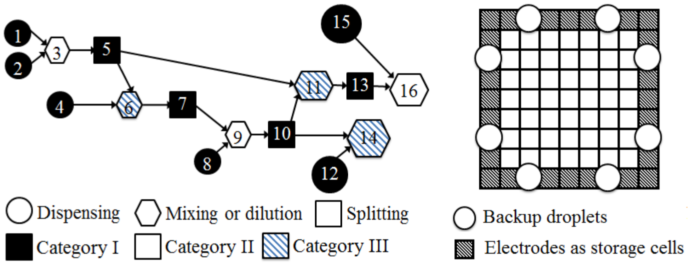
### B. Error Recovery Strategies

In this subsection, we formulate the principles underlying error recovery. For the given bioassay protocol, we use optical sensors to evaluate the quality of output droplets of each dispensing, mixing, dilution and splitting operation. According to the data provided by [3], [12], the response time of on-chip optical sensors are in the scale of picoseconds or nanoseconds. Thus the time cost for adding optical detection operations is negligible. Due to space constraints, the optical detection operations are not shown in the sequencing graphs in this paper.

For a microfluidic biochip, fluidic handling operations can be divided into two categories: reversible and nonreversible operations. Reversible operations include dispensing and splitting operations; nonreversible operations include mixing and dilution operations. For errors that occur at reversible operations, their recovery processes are relative simple. In a splitting operation, if two droplets with unbalanced volumes are generated, then the biochip will first merge the two abnormal droplets to a larger one and then split the larger droplet again. For errors that occur at a dispensing operation, the chip can send the abnormal droplet back to the corresponding reservoir and dispense another droplet. Thus for errors that occur at reversible operations, the time cost for recovery is small and no additional droplets need to be consumed.

The error recovery process for nonreversible operations is more involved. To implement the corresponding nonreversible operations to correct the error, we also need input droplets from operations whose outputs feed the inputs of the failed operation. Thus we may need to re-execute all the predecessors of the erroneous operation. For instance, if an error occurs at operation 7 in Figure 2(a), operations 1, 2, 3, 4, 5 and 6 may need to be re-executed. Thus the time cost for executing error recovery operations can be extremely high. To reduce the incidence of the worst case, the following strategies are taken in our approach:

- For a splitting operation, if only one of its output droplets is used as the input for the immediate successors, we store the other (redundant) droplet as a backup for possible error recovery at a subsequent stage. For example, operation 7 in Figure 2(a) is a splitting operation and it generates two output droplets. Only one of these two droplets is used as the input of operation 9. (Note here each circle in the sequencing graph stands for a fluidic handling operation. The unused droplets are not shown in the sequencing graph.) If an error occurs at operation 9, the redundant droplet will be used as an input for re-execution.



(a) (b)

Fig. 2. (a) An example of a sequencing graph corresponding to a bioassay protocol; (b) The layout of a biochip with reserved area for error recovery.

- All dispensing operations are scheduled for execution as early as possible and their output droplets are stored on the biochip. We also dispense some droplets as backup for possible error recovery operations. When the bioassay is completed, those unused backup droplets are sent back to the corresponding reservoirs.

Thus, when an error occurs at a nonreversible operation, the controller software first checks whether the inputs of this operation can be provided by backup droplets stored on chip. If the answer is yes, then the time cost for this operation can be reduced. Otherwise, more operations will be executed during error recovery. Based on the above discussion, the operations in the sequencing graph can be divided into three categories according to the number of operations and droplet consumptions in their error recovery processes, as shown in Figure 2(a). The first category includes all reversible operations and they can be simply re-executed when an error occurs; the second category includes nonreversible operations for which their immediate predecessors can provide backup droplets; all the other operations are placed in the third category.

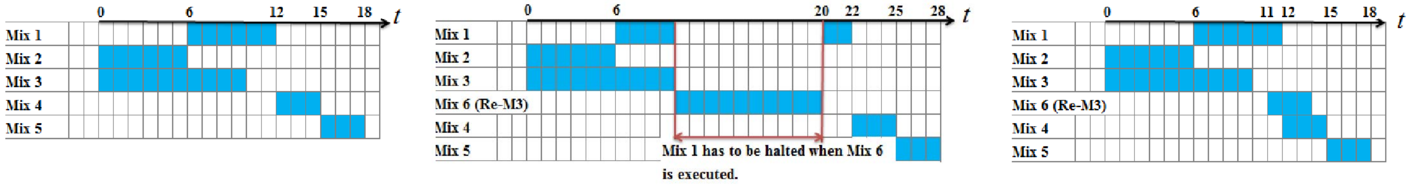
During the generation of the initial synthesis result, some electrodes are intentionally left unused and reserved for storage of backup droplets. An example is shown in Figure 2(b); all electrodes on the boundary of the chip are used as storage cells. Thus backup droplets can be easily transported on the biochip.

### C. Error Recovery

During bioassay execution, the control software needs to implement the following steps.

Step 1: Error identification. After each optical detector operation, the software will compare the detection result with a pre-assigned error threshold. If the optical detection result fails to meet the requirement of the experiment we conclude that an error has occurred. The control software will now begin to analyze the error.

Step 2: Error analysis. When an error occurs, the control software will begin to determine recovery operations. According to above discussion, if an error occurs at a reversible operation, the recovery process is simple. For non-reversible operations, the control software needs to search its predecessors until it finds a predecessor operation that can provide copy droplets to feed the inputs of the recovery subroutine. It is important to note that, for some operations, the recovery subroutines may change depending



(a) (b) (c)

Fig. 3. (a) Scheduling result when no error occurs; (b) Schedule when an error occurs in Mix 3. Mix 1 is halted when error operations are executed; (c) Schedule when an error occurs in Mix 3. Here dynamic synthesis strategy is applied at time 11 and error recovery operations begin at time 12.

on the error. For example, operation 7 in Figure 2(a) generates two droplets; one of them is used in the subsequent reaction and the other is stored on chip as the "copy droplet". If a single error occurs at operation 14, the biochip will re-execute operations 8, 9, 10 and 12. However, if an error occurs at a predecessor of operation 14, the recovery subroutine for operation 14 will be different. For example, when an error occurs at operation 9, the copy droplet of operation 7 will be used as the input for error recovery. If another error occurs afterwards at operation 14, there is no more copy droplet available at the output of operation 7. Thus the recovery subroutine of operation 14 has to be expanded and it will now include operation 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 12. Therefore the recovery steps are completely different from the case when an error occurs at operation 14.

After the recovery subroutine of an operation is determined, the control software will update the sequencing graph and the corresponding DAG, and then the dynamic re-synthesis step will be implemented.

Step 3: Dynamic Re-synthesis. In the cyberphysical system envisioned here, when an error is detected at a checkpoint, it will trigger the generation of a new mapping of the remaining steps (including proper handling of intermediate results) of the bioassay. This process is referred to as *re-synthesis*, on the basis of the initial design obtained from *a priori* synthesis.

The first step in the re-synthesis procedure is to determine all successor operations for the erroneous fluidic operation and delete these operations from the initial synthesis results. Other operations will be executed according to the initial synthesis result. Next the control software will determine all operations that need to be re-scheduled. These operations include error recovery operations and all successors of the erroneous operation.

Dynamic re-synthesis on the microfluidic array can be modeled as the *module placement with obstacles* problem. Here the operations that are implemented based on the initial synthesis result are fixed *a priori* as the "obstacles" while the other operations that are necessary for recovery are derived through re-synthesis and placed in the remaining available chip area.

Force-directed-relaxation (PDR) can be easily tailored to handle the *module placement with obstacles* problem [20]. However, the PDR method has two drawbacks: (i) it is a heuristic algorithm and it takes a relative long time to compute the module placement result; (ii) In our problem, the "obstacles" on the microfluidic array are dynamic and changing with time while the PDR is sensitive to the initial placement [21], [22]. Thus PDR is not suitable for the re-synthesis problem. Another straightforward method to solve the problem of *module placement with dynamic obstacles* is to use list scheduling and an exhaustive search method to derive the re-synthesis result.

We solve the *module placement with obstacles* problem as follows. The control software places all operations that need to be re-scheduled in a priority queue based on topological sort. The "deepest" operation in the subroutine (i.e., the operation at the bottom of the list generated by topological sort) is assigned the lowest priority while the "shallowest" (at the top of the list produced by topological sort) operation is assigned the highest priority in the queue. Next the control software needs to allocate on-chip resources to these operations. Here our on-chip resource set $R$ changes with time $t$. The control software will search for available resources at the current time for the operation with the highest priority. For example, if the operation with the highest priority is a mixing operation, then the system will search for an available $m \times n$ cell array that is not occupied from current time $t$ to $t$+$\triangle$t. Here $\triangle$t is the time needed for the operation in the $m \times n$ cell array. If suitable idle resources are available, resource binding will be successful and the start time of the operation will be deemed to be the current time. Otherwise, the operation has to be delayed until there are available resources. If multiple resources are available at the same time, the control software randomly chooses one and bind it to the corresponding operation. After resource binding and start/stop time of the operation are determined, the operation is removed from the priority queue.

The above steps are repeated to search for available resources until the priority queue is empty. The start and stop times for all operations that need to be adjusted are determined, and the re-synthesis process is completed. The control software also generates new actuation sequences according to the new synthesis results.

For a microfluidic biochip with an $M \times N$ electrode array, $D$ detectors and $P$ dispensing ports, the computational complexities of searching for available resources in this re-synthesis algorithm is O($MN + D + P$). Since we can view the number of detectors and dispensing ports as being constant and we are interested in algorithm scalability for large arrays, the worst-case complexity is O($MN$). This is because the software will exhaustively search each electrode/detector/dispensing port in the array and check whether it is available. The computational complexity for other parts of the algorithm are all O(1). Hence the computational complexity of the re-synthesis algorithm is O($MN$).

An example of re-synthesis is shown in Figure 3. Figure 3(a) shows the schedule corresponding to the sequencing graphs in Figure 1(a). Figure 3(b) and Figure 3(c) both show the schedules corresponding to the sequencing graph in Figure 1(b). Due to

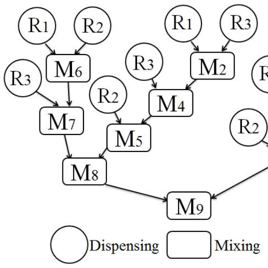


Fig. 4. Sequencing graph for sample preparation of plasmid DNA.

space limitations, we only show the schedule for mixing operations. Here Figure 3(b) is the schedule obtained using the error recovery algorithm of [11]. From Figure 3(b), we can see that mixing operation Mix 1 is halted for 10 time slots when error recovery operations are executed. The completion time of the bioassay shown in Figure 3(a) is increased from 18 time slots to 28 time slots, which can be unacceptable for many applications. The dynamic scheduling result corresponding to the sequencing graph in Figure 1(b) is shown in Figure 3(c). When the error is detected at the output of Mix 3 at time 10, the ongoing operation Mix 1 is executed based on the initial synthesis result. We assume that the computing time to generate the new synthesis result is 1 time slot. In practice, computation time is at least an order of magnitude less than the fluidic operation time. Then at time 11, the control software will generate new synthesis result based on the updated sequencing graph shown in Figure 1(b). As shown in Figure 3(c), Mix 1 is completed at time 12 without being interrupted and the experiment is finished at time 18. Thus the bioassay is executed “seamlessly” without any time penalty or interruption of other operations.

## IV. SIMULATION RESULTS

In this section, we evaluate the re-synthesis approach for error recovery on representative bioassays that are especially prone to fluidic errors. We evaluate the completion time for the re-synthesis approach and compare these results to the recently-proposed error recovery method [11] that does not utilize “physical aware” control software.

### A. Preparation of Plasmid DNA

First, we simulate the preparation of plasmid DNA by alkalinelysis with SDS-minipreparation [23]. During sample preparation, a mixture of three reagents is required. The three reagents are:

- $R_1$: Alkaline lysis Solution I (50 mM Glucose,25 mM Tris-HCl (pH 8.0), 10 mM EDTA (pH 8.0)).
- $R_2$: Alkaline lysis Solution II (0.2 N NaOH, 1% SDS (w/v)).
- $R_3$: Alkaline lysis Solution III (5 M sodium acetate, glacial acetic acid).

The required concentration of the mixture is 0.22% of $R_1$, 0.44% of $R_2$, and 0.34% of $R_3$, which can be approximated to $\frac{28}{128}$ of $R_1$, $\frac{56}{128}$ of $R_2$, and $\frac{44}{128}$ of $R_3$. The sequencing graph to get the required concentration by mixing $R_1$, $R_2$, and $R_3$ is shown in Figure 4. This bioassay is mapped to a $10 \times 10$ electrode array and all electrodes on the boundary of the array are used as storage cells.

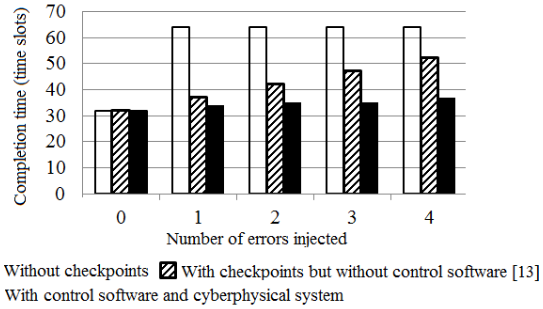


Fig. 5. Completion times for the three strategies when multiple errors are injected.

The error-recovery capability of the cyberphysical microfluidic system can be evaluated on the basis of the bioassay completion time when multiple errors are detected. We randomly inject a given number of errors into the chip during bioassay execution and compare the completion time of three strategies for error recovery.

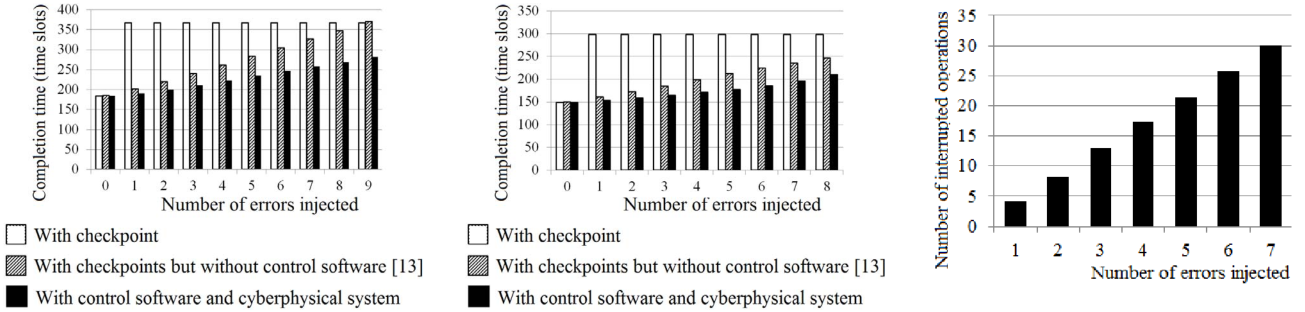
For the first strategy, when an error occurs during execution, the biochip has to re-execute the entire bioassay. In the second approach, as introduced in [11], when errors occur, all the error recovery operations are carried out in a stand-alone manner, i.e., all other bioassay-related fluidic operations are halted until recovery operations are completed. Based on published data from [11], the wait time is usually longer than the completion time for a single mixing/dilution operation, which is detrimental to the integrity of the biosassay outcome. For the third (proposed) strategy, when an error occurs, the cyberphysical system first stops all operations on chip and carries out re-synthesis for error recovery. The new synthesis results and actuation sequences are re-loaded into the memory of the controller, which activates the electrodes of the biochip. Following this step, all the fluidic operations resume. The wait time for the bioassay fluidic operations equals the computation time needed for re-synthesis, which is at least an order of magnitude less than the time needed for microfluidic operations.

As expected, Figure 5 shows that the cyberphysical microfluidic system leads to the shortest assay completion time in the presence of errors. We set the maximum number of errors to be four because it is representative of what a chip user is likely to accept. If more errors occur during execution, the chip is likely to be discarded and a new chip used. These results show that even with four errors during the bioassay, the proposed method leads to an increase of only 16% in the completion time compared to the case of no error. In contract, [11] leads to an increase of 63%.

### B. Protein Assays: Exponential Dilution and Interpolating Mixing

Next we evaluate the re-synthesis and error recovery approach for two real-life protein assays. These assays lead to the dilution of a protein sample using two methods, namely exponential dilution and interpolating mixing. The protocols and corresponding sequencing graphs for these two bioassays are described in [11].

Figure 6(a) and Figure 6(b) report the completion times when multiple errors are inserted during the *exponential dilution* and *interpolating mixing* protocols. Both bioassays are mapped to a



(a) Completion times for the three strategies when multiple errors are injected during exponential dilution.

(b) Completion times for the three strategies when multiple errors are injected during interpolating mixing.

(c) Number of operations interrupted when multiple errors are injected (interpolating mixing).

Fig. 6. Simulation results for two representative protein assays.

$12 \times 12$ electrode array and all electrodes on the boundary of the array are used as storage cells.

As discussed above, the proposed cyberphysical system approach reserves (allocates) a part of the array for potential error recovery during run-time. Figure 6(c) shows the number of operations that are interrupted when multiple errors are inserted in the interpolation mixing chip design and the algorithm of [11] is used. In the proposed cyberphysical system approach, no operations are interrupted and the bioassay operation is transparent to error occurrences.

## V. Conclusions

We have shown how recent advances in the integration of optical sensors in a digital microfluidics biochip can be used to make biochips error-resilient. We have presented a cyberphysical approach for "physical-aware" system reconfiguration that uses sensor data at intermediate checkpoints to dynamically reconfigure the biochip. A sensor-driven re-synthesis technique has been used to recompute electrode-actuation sequences, thereby deriving new schedules, module placement, and droplet routing pathways, with minimum impact on the time-to-response. The coordination between the physical-aware control software and the microfluidic biochip allows sensor data at intermediate checkpoints to be used as feedback to make decisions about completed operations, and dynamically reconfigure the biochip and optimize electrode actuation sequences for subsequent operations. The proposed approach has been evaluated and its effectiveness demonstrated for three representative protein bioassays.

## References

[1] H. Becker, "Microfluidics: a technology coming of age", *Medical Device Technology*, vol. 19, 2008.

[2] R. B. Fair et al., "Chemical and biological applications of digital-microfluidic devices", *IEEE Design & Test of Computers*, vol. 24, pp. 10-24, 2007.

[3] R. B. Fair, "Digital microfluidics: Is a true lab-on-a-chip possible?", *Microfluidics and Nanofluidics*, vol. 3, pp. 245-281, 2007.

[4] F. Su and K. Chakrabarty, "High-level synthesis of digital microfluidic biochips", *ACM Journal on Emerging Technologies in Computing Systems*, vol. 3, Article 16, January 2008.

[5] E. Maftei and P. Pop and J. Madsen, "Routing-based synthesis of digital microfluidic biochips", *Proceedings of the 2010 International conference on Compilers, Architectures and Synthesis for Embedded Systems*, pp. 41-50, 2010.

[6] T.-W. Huang and C.-H. Lin and T.-Y. Ho, "A contamination aware droplet routing algorithm for the synthesis of digital microfluidic biochips", *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 29, no. 11, pp. 1682-1695, November 2010.

[7] T.-W. Huang and T.-Y. Ho, "A two-stage ILP-based droplet routing algorithm for pin-constrained digital microfluidic biochips", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol 30, no. 2, pp. 215-228, February 2011

[8] M. Iyengar and M. McGuire, "Imprecise and qualitative probability in systems biology", *International Conference on Systems Biology*, 2007.

[9] O. Levenspiel, "Chemical reaction engineering", *New York: Wiley*, 1999.

[10] C. A. Mein et al., "Evaluation of single nucleotide polymorphism typing with invader on PCR amplicons and its automation", *Genome Res.*, vol. 10, pp. 330-343, 2000.

[11] Y. Zhao, T. Xu and K. Chakrabarty, "Integrated control-path design and error recovery in digital microfluidic lab-on-chip", *ACM Journal on Emerging Technologies in Computing Systems*, vol. 3, no. 11, August 2010.

[12] S. Cho, S. W. Seo, M. Brooke and N. M. Jokerst, "Integrated detectors for embedded optical interconnections on electrical boards, modules, and integrated circuits", *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 8, pp. 1427-1434, 2002.

[13] R. Evans R., L. Luan, N. M. Jokerst, and R. B. Fair, "Optical detection heterogeneously integrated with a coplanar digital microfluidic lab-on-a-chip platform", *Proc. IEEE Sensors Conf.*, pp. 423-426, Oct. 2007.

[14] N. M. Jokerst, L. Luan, S. Palit, M. Royal, S. Dhar, M. Brooke and T. Talmage, "Progress in chip-scale photonic sensing", *IEEE Trans. Biomedical Circuits and Sys.*, vol. 3, pp. 202-211, 2009.

[15] W. Bialek and O. William and Jose Nelson, "Protein dynamics and reaction rates: mode-specific chemistry in large molecules?", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 85, pp. 5908-5912, 1988.

[16] V. Srinivasan, V. Pamula, M. Pollack and R. B. Fair, "Digital microfluidic biosensor for multianalyte detection", *Technical Digest IEEE MEMS*, pp. 327-330, 2003.

[17] A. Furtado and R. Henry, "Measurement of greenuorescent protein concentration in single cells by image analysis", *Analytical Biochemistry*, pp. 84-92, November 2002.

[18] S. Rodriguez-Cruz, J. Khoury, and J. Parks, "Protein uorescence measurements within electrospray droplets", *Journal of the American Society for Mass Spectrometry*, pp. 716-725, June 2001.

[19] R. Sedgewick, "Algorithms in C: graph algorithms", *Addison-Wesley*, Chapter 23, 2001.

[20] H. Murata, K. Fujiyoshi and M. Kaneko, "VLSI/PCB placement with obstacles based on sequence pair", *Proc. International Symposium on Physical Design*, pp. 26-31, 1997.

[21] L. Sha and R. W. Dutton, "An analytical algorithm for placement of arbitrarily sized rectangular blocks", *Proc. 22th ACM/IEEE Design Automation Conf.*, pp. 602-608, 1985.

[22] A. Alon and U. Ascher, "Model and solution strategy for placement of rectangular blocks in the Euclidean plane", *IEEE Trans. on CAD*, vol. 7, no. 3, pp. 378-386, 1988.

[23] V. Ananthanarayanan and W. Thies, "Biocoder: a programming language for standardizing and automating biology protocols", *Journal of Biological Engineering*, vol. 4, no. 1, 2010.