Locking of Biochemical Assays for Digital Microfluidic Biochips

Sukanta Bhattacharjee[1], Jack Tang[2], Mohamed Ibrahim[3], Krishnendu Chakrabarty[3], Ramesh Karri[2]
[1]New York University, Abu Dhabi, [2]New York University, New York, NY, [3]Duke University, Durham, NC

***Abstract*—It is expected that as digital microfluidic biochips (DMFBs) mature, the hardware design flow will begin to resemble the current practice in the semiconductor industry: design teams send chip layouts to third party foundries for fabrication. These foundries are untrusted, and threaten to steal valuable intellectual property (IP). In a DMFB, the IP consists of not only hardware layouts, but also of the biochemical assays (bioassays) that are intended to be executed on-chip. DMFB designers therefore must defend these protocols against theft. We propose to "lock" biochemical assays through random insertion of *dummy mix-split* operations, subject to several design rules. We experimentally evaluate the proposed locking mechanism, and show how a high level of protection can be achieved even on bioassays with low complexity. We offer guidance on the number of dummy mix-splits required to secure a bioassay for the lifetime of a patent.**

I. Introduction

Microfluidic technologies are now entering a phase of rapid commercialization and deployment. One indicator of this is the recent FDA approval of the Baebies SEEKER, a digital microfluidic platform for medical diagnostics [1]. The chemicals, materials, and biochemical protocols required to realize a modern microfluidic system are becoming increasingly sophisticated and complex, making the task of designing such a system impractical for a single organization. Therefore, it is expected that the manufacture of microfluidic systems will begin to adopt a horizontal supply chain, where the holders of intellectual property (IP) that dictate a biochip's functionality send their designs to a third-party foundry for fabrication [2]. Such an approach mirrors the manufacturing model established by the semiconductor industry.

An undesirable side-effect of this manufacturing model is the potential for *untrusted* third-parties, who in the course of performing their intended duties, also steal IP or alter designs to modify the functionality of the end product. It is critical that designers of microfluidic systems prevent IP theft not only to prevent financial losses, but also to preserve the trust of end users. Grey market devices fabricated with lower quality may not perform to the same standard as authentic devices, which may lead to faulty operation. Given that microfluidic systems are commonly employed in mission-critical applications, this would lead to a severe erosion in trust.

One of the most promising microfluidic technologies being deployed today is based on digital microfluidic biochips (DMFBs). DMFBs operate according to a sequence of low-level control signals that are derived from the high-level biochemical assay (bioassay) specification, which forms the IP. The bioassay designer must provide this high-level specification to the foundry, but will then be susceptible to IP theft. To address the need for IP protection on DMFBs, this paper presents the concept of biochemical assay locking.

Our specific contributions are as follows:

1) We propose to lock biochemical assays through the insertion of dummy mix-split primitives.
2) We define new *bioassay-specific* security metrics to evaluate the effectiveness of the proposed scheme.
3) We analyze the key strength, which differs fundamentally from classical encryption and logic locking in that protocols are executed in the fluidic domain.
4) We validate the approach with experimental data on several biochemical assays, and show that little overhead is required to achieve satisfactory performance.

The rest of the paper is organized as follows: In Section II, we provide background information on biochemical assays and state-of-the-art DMFBs. In Section III, we present our proposed locking technique. We derive security metrics in Section IV and perform a detailed security analysis in Section V. We then show experimental results in Section VI, discuss some subtle details in Section VII, and conclude in Section VIII.

II. Background

Digital microfluidic biochips operate according to the principle of electrowetting-on-dielectric (EWOD): the modulation of contact angle between a droplet and a hydrophobic surface as a function of applied electric potential [3]. EWOD can be harnessed for the precise control of droplets on a DMFB array consisting of patterned electrodes. By properly sequencing voltages on adjacent elecrodes, operations such as mixing, splitting, and transport can be implemented, and these can in turn be used to construct complex biochemical assays.

The control signals used to drive a DMFB array are called actuation sequences, and are generated through a high-level synthesis flow [4]. The input to the synthesis flow is the biochemical assay to be executed on-chip, which is typically specified in the form of a directed acyclic graph (DAG). Nodes represent fluid operations and the edges represent dependencies. This forms one major component of the IP required to fabricate a functional DMFB. The output is the actuation sequence, which is a set of electrode activation patterns to be applied to the DMFB at a fixed rate (Fig. 1).

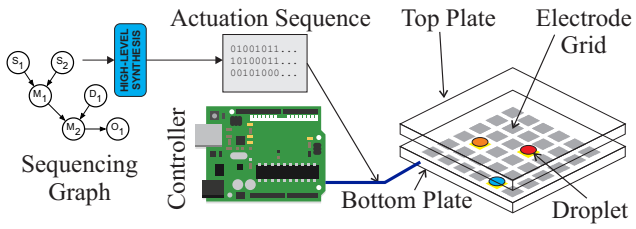

Fig. 1: Digital microfluidic biochips (DMFBs) use grids of electrodes to manipulate discrete droplets. The actuation sequence (set of control signals) is derived from a sequencing graph through high-level synthesis, and is sent to the DMFB from a controller unit.

978-1-5386-3728-9/18/$31.00 ©2018 IEEE

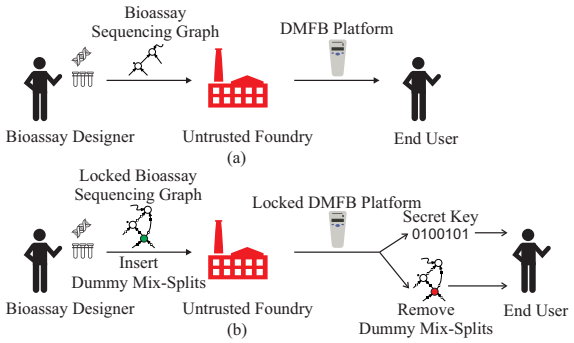

Fig. 2: (a) Untrusted DMFB platform design flow. The designer of the bioassay sends a sequencing graph to a foundry for fabrication. This forms the IP and can be stolen by untrusted foundries. (b) Proposed design flow. The bioassay designer inserts locking primitives that obscure the functionality of the design. The locked DMFB platform can be unlocked either through application of a secret key, or through removal of the inserted locking primitives.

Recently, the basic execution of actuation sequences on a DMFB have been extended to incorporate *conditional execution* [5], [6]. This is driven by the need for advanced biochemical protocols that alter their functionality depending on intermediate chemical reactions, and by the need for dynamic re-execution in case of run-time faults. This functionality will be leveraged in this work to unlock bioassays, which will be described in detail in Section III.

A. Untrusted DMFB Design Flows

We consider the DMFB design flow in Fig. 2(a). The design begins with the bioassay designer, or biocoder, who creates the biochemical assay and sends it to a third-party foundry for fabrication. The third-party foundry takes this bioassay, along with information on the fluids that the hardware must handle, cost and area constraints, and creates an integrated DMFB platform along with the synthesized actuation sequences. Such a manufacturing model offloads the burden of integrating the DMFB synthesis software with current hardware capabilities, which are subject to frequent change [7]. The completed DMFB platform is returned to the biochip designer, who can sell the platform to end users, or keep the platform for personal use. This custom design flow is in contrast to the general-purpose design flow that is often discussed in the DMFB design automation literature; in such works, it is assumed that the biochip designer can synthesize the actuation sequence and execute it on a programmable DMFB [2].

B. Related Prior Work

Security issues specific to DMFB platforms have recently been uncovered [8], many arising as a consequence of untrusted supply chains [2]. To counter IP theft, encryption of biochemical assays has been proposed at the *fluidic* level [9]. This approach uses a "fluidic multiplexer" (FMUX) as an encryption primitive that is inserted into the original assay. The FMUX selects between two input droplets for forwarding to the output depending on the presence/absence of a control droplet. The major shortcomings of this approach are that it can be broken through attacks executed in parallel, and that the output droplet must be mixed with an unspecified inert reference droplet during multiplexing. The design of the reference droplet is an open question, and would lead to incorrect droplet concentrations anyway. This is due to the inherent limitation of microfluidic logic gates [10] used to realize the fluidic MUX. Furthermore, implementation of the FMUX requires large chip area. This line of research takes some cues from the hardware security literature, where techniques identified as "logic locking" and "logic encryption" are used to protect VLSI designs from IP theft and unauthorized usage [11], [12]. We note that DMFBs are only one class of biochips, and that security issues are also being discovered in other design paradigms such as flow-based biochips [13], [14].

III. Bioassay Locking

We propose to lock bioassays by hiding true mix-split operations among randomly inserted dummy mix-splits. Conditional execution capabilities of state-of-the-art DMFBs are used to select which mix-splits to activate/deactivate. We target mix-split operations because they are abundant in nearly all bioassays, and they are critical for correct operation; if an attacker selects the incorrect mix-splits to activate/deactivate, then fluid outputs will be corrupted (Fig. 2(b)).

A. Preliminaries: Dummy Mix-Split

A dummy mix-split is a conditional mix-split operation that is not part of the original bioassay sequencing graph. A mix-split operation takes two input droplets, mixes them, and splits them into two output droplets of equal volume. A conditional mix-split operation is a mix-split operation that either mixes or does not mix two input droplets, based on some key value. We assumed that with key value 1, the mix-split operation stalls and then forwards the two input droplets to the two outputs without mixing them. With key value 0, the mix-split operation occurs normally. Mix-split operations can be implemented on the DMFB as a "virtual module," where a pre-defined number of electrodes are reserved for mixing. The two droplets to be mixed are routed to two virtual input electrodes, merged, and then routed around the virtual module for mixing. The mixing time is declared as part of the architectural specification of the DMFB platform. When mixing is complete, the droplets are split and sent to two virtual output electrodes for routing to subsequent operations [15]. The virtual mix-split module can be of variable size, such as 1×4 or 3×4 [16].

In a standard mix-split operation, the input and output electrodes are interchangeable. In a conditional mix-split, it is important that the input droplets are forwarded to the correct output port, otherwise the bioassay will no longer be correct. We introduce new symbolic notation to represent both standard mix-split and dummy mix-split operations, as shown in Figs. 3(c)-(d). We use two symbols (□ and ■) to identify the input and output ports. For example, in a 1×4 array mixer, the leftmost and rightmost cells can be used for inputs and outputs. This representation of mix-split operations helps the biocoder to hide the difference between the original and dummy mix-split operations in the locked sequencing graph.

B. Proposed Method

The process of locking a DAG proceeds as follows: the biocoder creates the bioassay, then replaces all mix-split op-

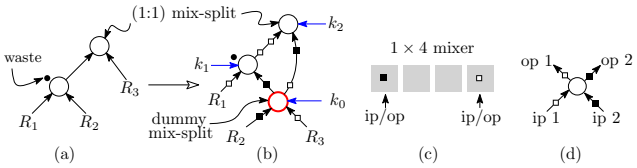

Fig. 3: (a) The bioassay specified as a sequencing graph. (b) Bioassay locking: Dummy mix-splits are added. A secret key dictates which mix-splits should be activated or deactivated. (c) A 1×4 linear mixer has two input/output (ip/op) ports which we denote with □ and ■. (d) The corresponding graph-level representation of the mix-split operation.

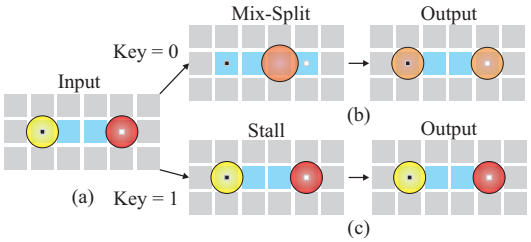

Fig. 4: (a) Input droplets for a 1×4 mix-split. (b) Conditional execution with key = 0 results in a standard mix-split operation. (c) Key = 1 results in a stall with no mixing.

erations with conditional mix-split operations with key value 0. Then dummy mix-splits are randomly inserted, which are deactivated with key value 1. The correct key values are kept secret. The locked DAG is sent to the foundry for synthesis of the actuation sequences and incorporation into a hardware platform. The dummy mix-splits are indistinguishable from real mix-splits, thus hiding the true functionality of the bioassay and preventing its unauthorized use. When the fabricated DMFB platform is returned, the end user must unlock the device by providing the correct key values for each mix-split operation. Alternately, they may remove the dummy mix-split operations. Note that this method does not pose any restrictions on existing synthesis algorithms, so the proposed modifications can be easily incorporated.

Example: Consider the input sequencing graph shown in Fig. 3(a), where two droplets of input reagents R_1 and R_2 are mixed together. After mixing, one of the two resultant droplets is mixed with a droplet of input reagent R_3. Before sending it to an untrusted design house, the input sequencing graph is locked by adding a dummy mix-split operation between R_2 and R_3. Note that for each mix-split node in the locked sequencing graph, each incoming edge is marked with a symbol (□ or ■) for differentiating the two different input/output ports on a mixer. Mixing operations are realized depending on the 0/1 value of the particular key bits. Without loss of generality, we assume that if the key value associated with a mixing operation is zero, two input droplets are mixed (Fig. 4(b)). Otherwise, two input droplets stall without mixing (Fig. 4(c)). The correct key for the example in Fig. 3 is $k_2k_1k_0 = 001$. After applying the correct key, the unlocked actuation sequence transports two droplets of R_2 and R_3 to the input ports of a mix-split. The mixing between two droplets of R_2 and R_3 is not performed as the key bit k_0 is set to one. However, the remaining two mixing operations are executed, as desired. Hence, the unlocked actuation sequence preserves the correctness of the bioassay.

C. Placement of the Dummy Mix-Splits

A dummy mix-split operation can be randomly inserted into a sequencing graph in several possible ways:

Add Extra Droplets: The simplest way is to add extra fluid droplets into a sequencing graph. Consider the sequencing graphs shown in Figs. 5(b)-(c), in which an extra input reagent droplet is added as a leaf node using a dummy mix-split operation. We have highlighted dummy mix-split nodes with a separate color and only relevant edges are distinguished with special symbols (□ or ■). In Fig. 5(b), we have to use a different input reagent (R_2 or R_3) from the other one (R_1) used in the dummy mix-split operation. However in case of Fig. 5(c), we can use any input reagents. If a wrong key is used to unlock the fabricated DMFB, undesired mixing with input reagent may take place.

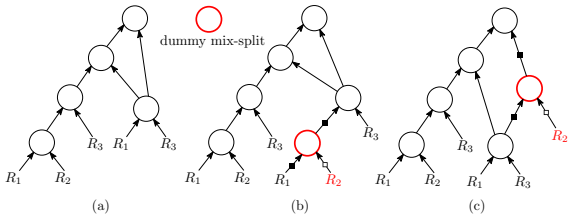

Fig. 5: (a) Input sequencing graph, (b)-(c) sequencing graph after inserting dummy nodes with extra input reagents.

Reuse Waste Droplets: A waste droplet available in the sequencing graph can be mixed with other intermediate droplets in a dummy mix-split operation. However, we cannot choose any arbitrary droplet because it may create a cycle in the locked sequencing graph. This is a design rule violation, as sequencing graphs with cycles are not synthesizable. For each waste droplet, we associate a subgraph of the sequencing graph on which the waste droplet is generated on the root node of that subgraph. We denote it as the "waste-subgraph" corresponding to the waste droplet. Fig. 6(a) shows the *waste-subgraph* for the waste droplet w_2. We use an intermediate droplet from subgraph, that is disjoint from the *waste-subgraph*, to participate in a dummy mixing node. In Fig. 6(b) the waste droplet w_2 and an intermediate droplet are used in a dummy mix-split operation.

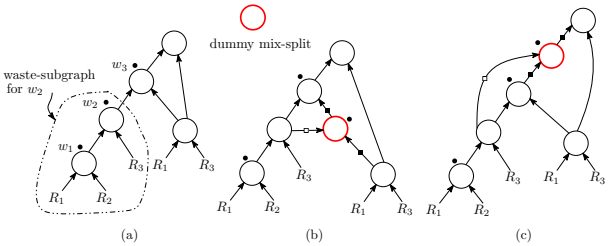

Fig. 6: (a) Input sequencing graph, (b)-(c) sequencing graphs after inserting a dummy node with a waste droplet as one input.

We may also combine a waste droplet with an intermediate droplet in a dummy mix-split operation that lies on the forward path starting from the root node of *waste-subgraph* associated

with the waste droplet. Fig. 6(c) shows the sequencing graph after inserting a dummy mix-split operation that combines waste droplet w_2 with an intermediate droplet.

Combine Two Subgraphs: Finally, we may combine two droplets from two different disjoint subgraphs of the input sequencing graph in a dummy mix-split operation. If a wrong key is used, undesirable mixing between the fluids represented by the two subgraphs is carried out, thereby corrupting the assay outcome. We have adopted a graph traversal technique for selecting candidate subgraphs. We start from an arbitrary leaf node (i.e., input reagent) and start traversing in the forward direction. If two disjoint subgraphs are found, we can use them in a dummy mix-split node. Otherwise, we start traversing from another leaf node in the sequencing graph.

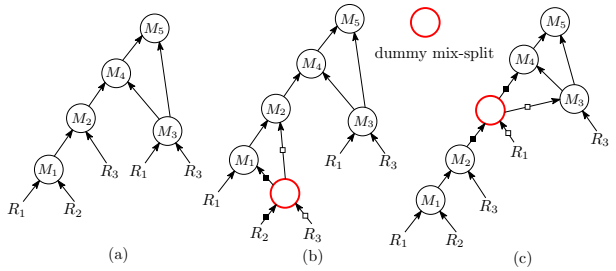

Fig. 7: (a) Input sequencing graph, (b)-(c) sequencing graphs after inserting a dummy node that use droplets from two different and disjoint subgraphs.

Fig. 7(b) is generated by combining two droplets of R_2 and R_3 in a dummy mix-split node, and these two different and disjoint subgraphs (consist of a reagent node only) are found by following the path $(R_1, M_1, M_2, M_4, M_5)$ in the sequencing graph shown in Fig. 7(a). Analogously, Fig. 7(c) combines droplets corresponding to two left subgraphs of M_3 and M_4 that lie on the path (R_3, M_3, M_4, M_5) of the sequencing graph shown in Fig. 7(a).

D. Implementation Details

The strength of the proposed method is that unlocking is fast and simple: an end user loads a digital secret key into a small tamper-proof memory module. The weaknesses are that the DMFB must prevent observation of the fluid movements, and that dummy mix-splits introduce stalls into the assay. Stalls may be unacceptable for assays with strict completion time requirements, but slows down brute-force attacks. The ease of use makes this method well-suited for applications where one can sell the secret keys to end users.

If stalls or tamper-proof memory are not acceptable, one may alter the synthesized actuation sequence to trim out the dummy mix-split operations. Methods developed for the adaption of synthesized actuation sequences for DMFB technology change can be used for the automated removal of the dummy mix-split operations [7]. It is more computationally efficient to change a synthesized actuation sequence than to generate it from scratch, thus preserving one motivation for outsourcing DMFB fabrication.

Stall removal requires that the end user process the synthesized actuation sequence. A trusted third party should provide the software. The actuation sequence provided by the foundry needs to be accessible and modifiable. This is a more complex usage scenario, but has the advantage of recovering the original assay which can be executed with zero overhead. However, an attacker who gains physical possession of the DMFB platform can extract the unlocked actuation sequence and thus reverse engineer the bioassay. So, this method is better suited to private users who will not relinquish physical control of the unlocked DMFB, such as researchers developing novel bioassays.

IV. Security Metrics

A locked design, upon application of an incorrect key, must produce an output that is as dissimilar from the correct output as possible. To do otherwise would be to either defeat the purpose of locking (e.g., producing an output that is approximately close enough) or leak information that can be leveraged to recover the original bioassay. Fluids possess a multitude of physical properties with interactions that are difficult to accurately model. Therefore, we believe that *security metrics for bioassay locking are bioassay-dependent.* This is a significant departure from VLSI logic locking, where security metrics are circuit agnostic. We expect that a multitude of security metrics will be discovered for related families of bioassays. For this work, we focus on sample preparation bioassays and therefore define the first bioassay security metric in terms of output ratios.

A. Preliminaries: Sample Preparation

In many bioassays, a mixture of reagents must be generated to meet a specified concentration ratio. This process is called sample preparation [17], [18], [19], and in a DMFB it is typically implemented by repeatedly mixing two droplets of equal volume and splitting the resultant droplet into two equal size droplets (i.e., using a 1:1 mix-split operation [17], [18]).

We consider sample preparation assays that take k reagents $R_1, R_2, \cdots, R_i, \cdots, R_k$ and generates a mixture with ratios of $O = \{c_1 : c_2 : \cdots : c_i : \cdots : c_k\}$ where c_i denotes the corresponding concentration factor (*CF*) of reagent R_i. A pure reagent has $CF = 1$ while neutral buffers have $CF = 0$. Since O is a mixing ratio, it must satisfy $\sum_{i=1}^{k} c_i = 1$ [19]. A sample preparation assay may generate m different outputs, which we denote as a set $T = \{O_1, O_2, \cdots, O_j, \cdots, O_m\}$, where each O_j specifies a different mixture of reagents.

B. Output Ratio Corruption

When an incorrect key is applied to the locked bioassay, we desire a large number of outputs to be corrupted beyond some error tolerance ϵ. We therefore define our security metric as *the proportion of outputs whose ratios exceed the error tolerance*. We call this *output ratio corruption (ORC)*. We can determine whether an individual output is corrupted by measuring the uniform norm of the difference between an output and its specification. If this norm exceeds ϵ, it is corrupted. We define a helper function to indicate corruption as:

$$\phi(j) = \begin{cases} 1 & ||\hat{O_j} - O_j||_\infty > \epsilon \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where we use the uniform norm defined as

$$||\hat{O_j} - O_j||_\infty = \max\{|\hat{c_1} - c_1|, \cdots, |\hat{c_k} - c_k|\} \quad (2)$$

The hat $(\hat{O_j}, \hat{c_k})$ indicates the correct ratio values and (O_j, c_k) indicate the actual ratios for some incorrect key. Then, the output ratio corruption is measured as

$$ORC = \frac{100}{m} \sum_{j=1}^{m} \phi(j) \tag{3}$$

In other words, this is the percentage of outputs that are corrupted. Ideally it is 100% for all possible incorrect keys. This is different than logic locking, which ideally has 50% corruption as measured using the Hamming distance metric [11]. Digital outputs that are 100% corrupted are simply the complement of the correct output. No such notion exists for fluid concentrations.

V. Security Analysis

A. Brute-Force Attacks

A brute-force attack attempts to recover the original bioassay by trying all key combinations. This is equivalent to the problem of identifying which mix-split operations are dummy mixers. Assume the original assay contains n mixing operations. If we insert d dummy mixing operations, the locked assay will contain $n + d$ mixing operations. If the attacker knows d, then they must consider $\binom{n+d}{d}$ different combinations of dummy mixers to remove. If the attacker has no knowledge of d, then they must consider every possible combination of dummy mixers to remove, up to $d = n$. This is because $\binom{n+d}{d}$ is maximized for $d = n$. Therefore the total number of subsets to consider is 2^{n+d-1}. Only the empty set is invalid, so finally the total number is $2^{n+d-1} - 1$.

B. Key Length Selection

The key length determines the strength of the locking and is dictated by the number of parallel experiments an attacker can execute p, the bioassay execution time m, and the required minimum lifetime of the protection λ. If the attacker knows d, the number of dummy mixers required must satisfy the inequality

$$\binom{n+d}{d} \geq \binom{\lambda \cdot p}{m} \tag{4}$$

where the left side represents the number of brute-force attacks required to break the locking scheme, and the right side is the number of bioassays that can be executed in a given time period. If the attacker does not know d, then the quantity on the left side becomes 2^{n+d-1}, which leads to the inequality

$$n + d \geq \log_2 \left(\frac{\lambda \cdot p}{m} + 1 \right) + 1 \tag{5}$$

If we assume $\lambda = 20$ years (the lifetime of a US patent), $p = 1000$, and $m = 1$ minute, the right side quantity is equal to 34.3. That is, as long as the total number of mix-split operations is greater than 34, enough security can be achieved for patent protection. This implies that small bioassays can be secured by adding a large number of dummy mix-split operations, while large bioassays require less. However, in practice, these parameters may vary: the bioassay execution time is variable and doesn't include the time required to prime the DMFB platform and interpret results, while the cost to manufacture a hardware platform will deter parallel attacks.

VI. Experimental Results

We implemented the proposed locking scheme in Python 2.7 and evaluated it on several benchmark DMFB assays: PCR mixture droplet streaming [7], PCR mixture preparation [17], multi-target dilution [18], and protein assay [7]. Locked assays were synthesized using MFStaticSim [20] with list scheduler, left-edge placer, and modified maze router. We assume mix-splits take four seconds on a 2 Hz DMFB. Note that the output of the protein assay is not a set of target ratios, and therefore output ratio corruption is not applicable. Nevertheless, we include results on its overhead.

A. Output Ratio Corruption

We inserted five dummy mix-splits into the assays to show that acceptable corruption can be achieved with low overhead. We randomly selected a large number (1000) of input keys and compared the corresponding output ratios against the correct outputs with the output ratio corruption metric. For the PCR mixture droplet streaming assay, we set $\epsilon = 1/32$. We plot the histogram of ORC values in Fig. 8(a), and observe the values are concentrated at 100%. The multi-target dilution assay was evaluted with $\epsilon = 1/128$. We also observe the same concentration of values near 100% (Fig. 8(b)). For both of these assays, no outputs with 0% corruption were observed. The PCR mixture preparation assay has only one output, which we observed as always being corrupted when tested with $\epsilon = 1/256$. Note that the error limit ϵ was used in generating the source sequencing graphs. While this evidence suggests that bioassay locking can ensure that no random guess gives a correct output, we have not provided theoretical guarantees that this is true. This will be addressed in future work.

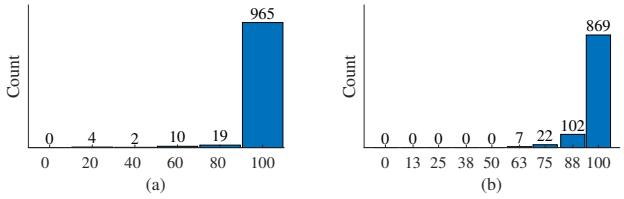

Fig. 8: Histograms of Output Ratio Corruption [%]. 1000 random keys were used to unlock the assays. (a) PCR mixture droplet streaming. (b) Multi-target dilution.

B. Overhead

We quantify the overhead in terms of number of dummy mix-splits and assay execution time. The number of inserted dummy mix-splits correlates linearly with chip area overhead, as a mixing module must be reserved on-chip. As seen in the evaluation of ORC, the outputs are sufficiently corrupted even with a small number of dummy mix-splits, so the overhead is small. The overhead in terms of assay execution time is shown as a function of number of dummy mix-split operations in Table I. In some cases the locked assay time can increase by over 50%. However, as shown in Section V-B, only a small number of dummy mix-splits can be used to secure larger assays. The protein assay only increases by 16% if only five dummy mix-splits are inserted. If it is desired to lock an assay with strict scheduling requirements, then the DMFB designer can remove the stalls as described in Section III-D.

TABLE I: Variation of assay execution time with respect to d, the number of inserted dummy mix-split operations.

Assay	Mix-Splits n	Execution Time [s]	DMFB Size	Locked Assay Execution Time [s] $d = \{$ 4	5	6	7	8	9	10 $\}$
PCR mixture droplet streaming [7]	15	35	16 × 15	41	47	49	57	58		
PCR mixture preparation [17]	19	47	16 × 15	57	67	67	65	71		
Multi-target dilution [18]	28	67	15 × 15		69	77	69	79	91	93
Protein assay [7]	47	75	15 × 19		87	91	93	95	91	89

VII. Discussion

We highlight subtle aspects of this work in a Q&A format:

Q1. *Why can't an attacker just simulate all the outcomes?*

Ans: Technically, they can, but they will not be able to tell whether or not the result is correct without synthesizing the outputs and using them in a biochemical assay. If the attacker could tell what the correct output is, then they could simply synthesize it without having to unlock the bioassay.

Q2. *Can an attacker collect several partially correct outputs and use them to reconstruct a complete output?*

Ans: No, assuming the outputs are to be used together in some target application. If the outputs can be used independently, then the locking scheme would require that all outputs are corrupted for every incorrect key.

Q3. *Could an attacker acquire an unlocked platform and reverse engineer the bioassay by measuring the concentrations?*

Ans: It depends. While we have considered output ratio corruption as the security metric, the underlying concentration factors are only "measurable" because we know how each output droplet was created. There does not exist a generalized method to detect the concentrations of the constituent reagents in a droplet. Additionally, some reagents may interact and form new compounds, complicating the situation. Therefore, we assume that reverse engineering of the outputs is not feasible.

Q4. *How is this work similar/different from logic locking?*

Ans: They are similar conceptually: by inserting some locking primitive into a design, an attacker loses both the ability to discern the functionality and use it in a black-box fashion. They differ in that, in bioassay locking, the original design elements (mix-split operations) are identical to the inserted locking primitive. The attacker's dilemma is to determine how many and which mix-split nodes to remove. This would be analogous to being presented with a VLSI gate-level netlist and being asked which gates to remove. Logic locked VLSI designs have easily identifiable key gates. The attacker's dilemma is to figure out whether the key gate should be driven by a 0 or a 1. Furthermore, locked bioassays cannot be easily simulated and checked, and brute-force attacks are time and cost constrained.

VIII. Conclusion

We have presented the first *practical* biochemical assay locking scheme for DMFBs. We leverage dummy mix-split operations and conditional execution as a locking primitive, and show that it is possible to achieve strong performance even with small key sizes. This approach is easy to implement and with some overhead in chip area and execution time. We have also described a method to eliminate this overhead. Compared to previously reported fluidic locking techniques, this work provides strong key strength without any dependencies on inherent DMFB failure modes, and avoids the severe chip area penalty required to implement a fluidic multiplexer. We defined biochemical assay security metrics in terms of output ratio corruption. In future work, we plan to generalize this concept to include other properties such as volume and temperature.

Acknowledgments

This research is supported in part by the Army Research Office under grant number W911NF-17-1-0320, NYU Center for Cyber Security (CCS), and CCS-AD.

References

[1] Baebies, Inc. (2017) Baebies SEEKER. [Online]. Available: http://baebies.com/products/seeker/

[2] S. S. Ali *et al.*, "Supply-chain security of digital microfluidic biochips," *Computer*, vol. 49, no. 8, pp. 36–43, 2016.

[3] M. Pollack, A. Shenderov, and R. Fair, "Electrowetting-based actuation of droplets for integrated microfluidics," *Lab. Chip*, vol. 2, no. 2, pp. 96–101, 2002.

[4] F. Su and K. Chakrabarty, "High-level synthesis of digital microfluidic biochips," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 3, no. 4, p. 1, 2008.

[5] M. Ibrahim, K. Chakrabarty, and K. Scott, "Synthesis of cyberphysical digital-microfluidic biochips for real-time quantitative analysis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 5, pp. 733–746, 2017.

[6] D. Grissom, C. Curtis, and P. Brisk, "Interpreting assays with control flow on digital microfluidic biochips," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 10, no. 3, p. 24, 2014.

[7] S. Bhattacharjee *et al.*, "Adaptation of biochemical protocols to handle technology-change for digital microfluidics," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 3, pp. 370–383, 2017.

[8] S. S. Ali *et al.*, "Security assessment of cyberphysical digital microfluidic biochips," *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 13, no. 3, pp. 445–458, 2016.

[9] S. S. Ali *et al.*, "Microfluidic encryption of on-chip biochemical assays," in *Proc. Biomed. Circuits Syst. Conf.*, Oct. 2016, pp. 152–155.

[10] Y. Zhao and K. Chakrabarty, "Digital microfluidic logic gates and their application to built-in self-test of lab-on-chip," *IEEE Trans. Biomed. Circuits and Systems*, vol. 4, no. 4, pp. 250–262, 2010.

[11] J. Rajendran *et al.*, "Fault analysis-based logic encryption," *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 410–424, 2015.

[12] M. Yasin *et al.*, "On improving the security of logic locking," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 9, pp. 1411–1424, 2016.

[13] J. Tang *et al.*, "Security implications of cyberphysical flow-based microfluidic biochips," in *Proc. IEEE Asian Test Symp.*, Taipei, Taiwan, 2017, pp. 110–115.

[14] J. Tang *et al.*, "Security trade-offs in microfluidic routing fabrics," in *Proc. IEEE Int. Conf. Comput. Des.*, Newton, MA, Nov. 2017, pp. 25–32.

[15] D. T. Grissom and P. Brisk, "Fast online synthesis of digital microfluidic biochips," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 3, pp. 356–369, 2014.

[16] P. Paik, V. K. Pamula, and R. B. Fair, "Rapid droplet mixers for digital microfluidic systems," *Lab. Chip*, vol. 3, pp. 253–259, 2003.

[17] S. Roy *et al.*, "Layout-aware mixture preparation of biochemical fluids on application-specific digital microfluidic biochips," *ACM Trans. Design Autom. Electr. Syst.*, vol. 20, no. 3, pp. 45:1–45:34, 2015.

[18] S. Bhattacharjee, A. Banerjee, and B. B. Bhattacharya, "Sample preparation with multiple dilutions on digital microfluidic biochips," *IET Comput. Digit. Tech.*, vol. 8, no. 1, pp. 49–58, 2014.

[19] S. Bhattacharjee *et al.*, "Dilution and mixing algorithms for flow-based microfluidic biochips," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 4, pp. 614–627, 2017.

[20] D. Grissom *et al.*, "An open-source compiler and PCB synthesis tool for digital microfluidic biochips," *INTEGRATION, the VLSI journal*, vol. 51, pp. 169–193, 2015.