Int. J. Nanosci. Nanotechnol., Vol. 16, No. 3, Sept. 2020, pp. 167-179

Fault Tolerant DNA Computing Based on Digital Microfluidic Biochips

Maryam Isvandi

Department of Computer Engineering, Lorestan University, Khorramabad, Iran.

(*) Corresponding author: Isvandi.m@lu.ac.ir
(Received: 08 December 2019 and Accepted: 12 June 2020)

Abstract

Historically, DNA molecules have been known as the building blocks of life, later on in 1994, Leonard Adelman introduced a technique to utilize DNA molecules for a new kind of computation. According to the massive parallelism, huge storage capacity and the ability of using the DNA molecules inside the living tissue, this type of computation is applied in many application areas such as medical and engineering. Despite these advantages, DNA computing fault is prone to error. These errors may affect the entire computation and lead error in final result. Design of tolerant systems is one of the hot topics in the field of circuit design. The error in DNA computing will appear by a change in the concentration in compare to a threshold. In this paper, a buffer to modify the level of concentration is introduced and the number of required buffers in order to reduce the overhead caused by additional buffers in system is investigated using normal distribution. Designed system will modify any error with 15% changes in output concentration level in compare to a threshold level using the proposed method, which will increase the reliability.

Keywords: *DNA computing, Microfluidic biochips, Fault tolerant, Reliability, DNA circuit.*

1. INRODUCTION

DNA molecules have been known as the basic blocks to store the codes for protein generation and transfer the genetic features through the generations, but in recent decades, it is found that DNA molecules can be used for a new kind of computation [1]. Leonard Adleman utilized the DNA molecules to solve the Hamiltonian path problem [2]. Adleman's experiments show that complex problems can be solved using DNA molecules faster than silicon-based computers, thanks to the significantly huge degree of parallelism. DNA computation made a bridge between computer engineering, biological and biomedical sciences. The potential applications of this kind of processing include medical therapeutics, pharmacy, solving NP-Complete/NP-Hard problems and Gen analysis [3] and [4]. The design methods of DNA nanostructures have been developed over the last two decades to realize well-ordered DNA lattices to organize and control matter at the Nanoscale [5]. The following is a brief review of the literature and previous work on DNA-based logic gate design. Livstone et al. proposed a method of implementation for DNA-based AND/OR functions [6]. They claimed that their proposed structure, consisting of micro-reactors along with attached heating elements towards controlling DNA annealing process, is capable of solving the satisfiability problem in linear space and quadratic time [6]. Sakamoto et al. proposed the concept of localized DNA strand displacement, which is a mechanism for implementing chemical reaction network on a surface of a DNA Nano-structure [7]. Localization increases the relative concentration of strand reaction thus speeding up the kinetics [8]. Qian et al. proposed the Seesaw logic gates in 2009 [9] and further improved in 2011 [10]. Their methodology applies a number of strands to allow for increased scalability compared with earlier methods. Fan et al. in [11] proposed a three-input label-free/enzyme-free majority gate through DNA hybridization without DNA

replacement and enzyme catalysis; further, the system is capable of implementing various basic/cascade logic gates. Recently, DNA logic systems have been utilized successfully to detect the risky patterns of nucleotide based cancer biomarkers (microRNAs) [12] and [13]. Authors of [12] utilized the DNA logic gates as modules to create molecular computers with biological inputs. Modular circuits that recognize microRNA cancer biomarkers through strand hybridization activate computation cascades to produce controlled outputs. Microfluidic biochips provide the controlled and automated platforms for performing various biochemical procedures [14]. These chips are known as a promising platform for executing the DNA operations in a controlled process. In [15] discusses a flexible configuration platform for performing a DNA computation on a microfluidic architecture in order to realize basic logic structures such as switches, memories and logic gates; Their proposed design is capable of programming DNA strands into various Boolean problems. However, each technique comes with its own benefits and drawbacks as follows. A configurable DNA Architecture (DENA) and corresponding design methods are proposed in [16] to improve the feasibility and cascade-ability of DNA circuits that provides the basic concepts of configurable DNA architectures. However, this paper does not used real system. In fact, microarchitecture in [16] is used ideally and free of any fault. Therefore, electrodes can be used without any condition. But in reality this may never happen. In the real implementation various fault may occur in the processing system based on the DNA computing. Errors in DNA computing can be expressed as a reduction in concentration (Increase is rarely possible). if concentration decreases slightly, then it does not affect the output logic. with the increase in errors (concentration decreasing), the logic of the output may be changed. In this paper DNA computing implemented on microfluidic biochip. In fact, we used DENA architecture for DNA computing but not ideally. This paper is focused on the occurrence of faults that lead to reduce the output concentration. These faults are evaluated with two methods and some solutions are presented to overcome them. While the concentration reduction partly affectless on output logic, the output logic may be error-prone as concentration increases. The main concepts are addressed in the following section which will assist the reader to arrive at a better understanding of other sections.

1.1. DNA Logic Circuits

As mentioned before, operation of a DNA circuit relates to the reactions occurred between DNA strands based on Watson-Crick complementary rules [17]. Each DNA strands composed of a finite set of Nucleotides; Adenine (A), Guanine (G), Thymine (T) and Cytosine (C). In DNA computing everything is encoded in the form of strands of four alphabets A, T, G, and C [18].

Various styles are proposed for DNA-based logic gate design. However, two design styles are more popular; Toehold-mediated design style and Seesaw design style. These two design styles are described in the following subsection.

1.2. Toehold-Mediated Logic Gate Design Style

Yurkeet.al in [19] reported an interesting DNA hybridization reaction that realized the DNA logic gate implementation with better correctness and speed rather than existing methods. Their contribution is based on the fact that the merging probability of two complement DNA strands has a reverse dependency to their length. In other words, smaller complement strands have more merging probability than larger strands.

Authors of [19] used a small-length strand to accelerate the merging flow of the DNA strands and he called this strand

as Toehold that analogue to the primer in biology sciences. Using the Toehold strands, improves the controllability of DNA displacement process and hence reduces the DNA process time. A simple example of this method is shown in figure 1. In this figure, each green arc line shows a single DNA strand whose direction represents its merging direction (from 5' to 3'). Labels of complement strands are represented by quote sign (*) and also Toehold strands are highlighted by red color.

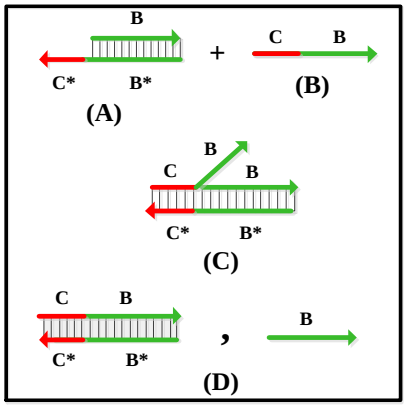

Figure 1. *Toehold mediated strand displacement.*

Figure 1-A shows a double strand that is generated by binding of strands B and B'C'. This double strand reacts with strand BC in which C is toehold strand (Figure 1-B). At first, Toehold C is connected to its complementary strand C' (Figure 1-C) and then strand B of BC strand replace with strand B in initial double strand. The resulting output is a single strand B and a double stranded BC (Figure 1-D).

This process is termed toehold-mediated strand displacement. Typical toehold lengths used for toehold-mediated strand displacement hybridization reactions range from 3 to 7 nucleotides.

Toehold-mediated strand-displacement enables the implementation of the DNA circuit without any enzymes for reactions, so this method is cheaper and faster than enzyme-based method. The drawback of this method is that many orthogonal strings are necessary for large circuit design to avoid unwanted reactions. Orthogonal DNA strands are defined as the strands that cannot be merged to gather because of their nucleotides' order.

Attachment interest of strand for displacement, which called strand kinetic, depends on the length and sequence of the toehold domain. This factor determines the speed of toehold mediated strand displacement reactions. Figure 2 shows the dependency of kinetics to length and type of nucleotides. In this figure, green graph shows the kinetics of using a maximally strong toehold composed of only G/C nucleotides, the red trace shows the kinetics of using a toehold composed only of A/T nucleotides, and the black traces shows the kinetics of a toehold composed of roughly equal numbers of all 4 nucleotides [20]. In addition to the type of nucleotides, number of them are effective.

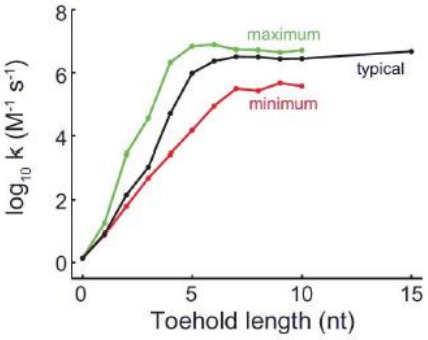

Figure 2. *Summary of strand displacement rate constants plotted against the invading toehold length n[20].*

Moreover, according to the [19] and [20], the DNA reaction rate will be increased when the number of toehold domains in a single strands grows. In addition, domain size is effective in reaction rate and choice between 4 – 7 nucleotide [20].

1.3. Seesaw Logic Gate

Qin and Winfree proposed the Seesaw logic design style at 2009 [14]. A seesaw logic gate comprises of five kinds of strands; inputs, outputs, gate, threshold and fuel in which output strand is produced by

interaction between input strands with gate and threshold strands.

Seesaw gates are designed based on the toehold-exchange protocol. Input strands act catalytically, hence a single input strand can help release multiple output strands from multiple seesaw gates. The output strands act as inputs for the next seesaw gates downstream in the circuit.

A seesaw gate with n-input and m-output, consists of n input strands, 3 internal strands and m output strands. Internal strands include gate, threshold and fuel strands.

Figure 3 shows a simplified view of a single-input and single-output seesaw base component. The input strand (S2.T.S5) consists of sub-strands S2, T and S5 in which T is toehold. Output strand (S5.T.S6) consists of sub-strands S5, T and S6. Gate strand is a composite strand which consists of upper strand S5, T and S6 and lower strand T*, S5* and T* (a* is complementary of a) that S5, T and S5*, T* composed of a double strand. Threshold strand includes s2* which is a small portion of starting bases of S2 and sub-strand T* and double strand S5.S5*. Fuel strand is a lower strand and consist of three domains; S5*, T* and S7*. Fuel strand increases the chance of reactions and probability of correct output production. Red numbers within the nodes or on the wires in Figure 3 indicate relative concentrations of different initial DNA species. Negative concentration used for threshold due to input consumption.

As shown in Figure 3-B, input strand reacts with both gate and threshold strands simultaneously. Input and threshold reaction rate is higher than input and gate reaction rate that cause to output strand production, due to the two toehold (s2* and T*) that described in Part I-B. Therefore, threshold concentration is an effective parameter in output concentration control. In the other words, the threshold gate can be thought as a garbage collector that makes a strand unusable for further reactions. By using the required number of seesaw base components (e.g. Figure 3) and adjusting the concentration of the threshold gate, both "AND" and "OR" gates can be constructed [15]. The main advantage of Seesaw logic gates is cascade-ability. This advantage is achieved due to slight loss of output concentration levels.

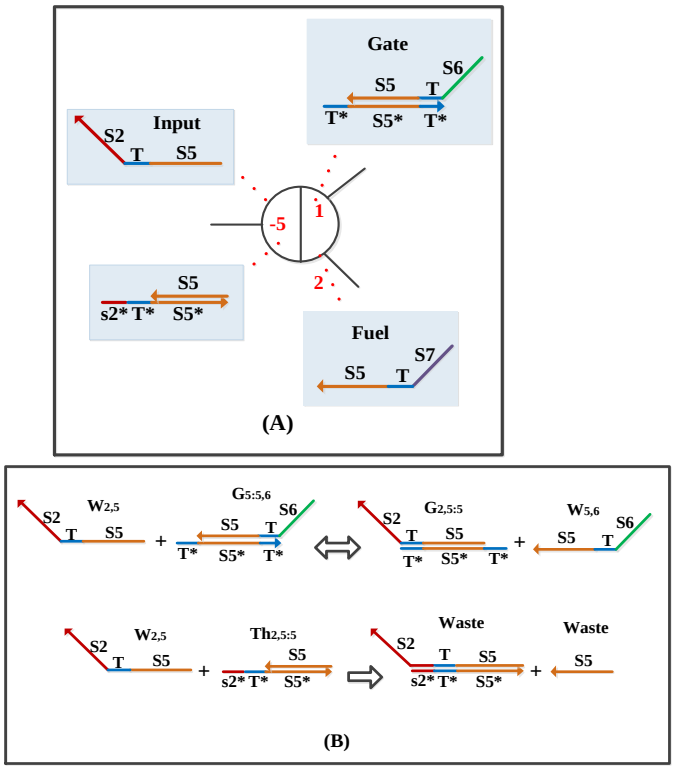

Figure 3. *Details of strands and operations in the seesaw base component system [15].*

1.4. Digital Microfluidic Platforms for DNA Computing

A digital microfluidic biochips (DMFB) device is a platform for performing operations of biological assays in an automatic and controllable manner [21]. These types of chips are comprised of a small bed made of paper, glass, plastic or silicon on which various biological operations towards analyzing human or animal chemical samples, manipulating and detecting samples are performed. A DMFB can manipulate discrete droplets of liquids on the surface of a two-dimensional array of electrodes; the actuation of electrodes can be programmed and controlled through software-driven

electronic control unit. A typical digital microfluidic biochips (DMFB) device is comprised of two top and bottom plates; the top plate is a single continuous electrode, whereas the bottom plate consists of an array of electrodes. Both plates are covered with a dielectric layer and further a thin hydrophobic layer in order to allow actuation of droplets and prevent adhesion of droplets to electrodes. The droplets are sandwiched between the gap of the top and bottom plates; the gap is typically filled with air or a filler fluid such as silicon oil so that actuation of droplets on the array of electrodes is further facilitated. Figure 4 illustrates structure of EWOD-based digital microfluidic biochips. One of the main drawbacks of DNA computing has been historical that chemical reactions between the DNA strands are manual operations and do not seems to be automated easily. Recently, considerable research is reported on using the DMFBs as the execution platform of DNA reactions.

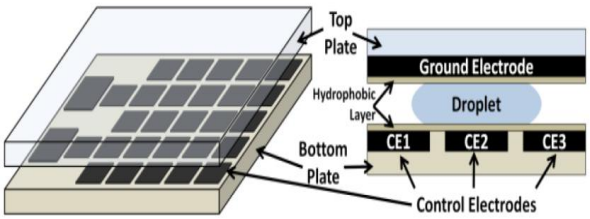

Figure 4. *Digital microfluidic biochip [21].*

1.5. DNA-Based Micro-Architecture

A digital A micro-architecture defines the building blocks and the configuration scheme of the components (e.g. Resources and methods) to realize the target circuit. An efficient micro-architecture is a necessary requirement for any automatic or semi-automatic design and large scale DNA circuit design. In [16], a micro-architecture (DENA) has been proposed enabling the large-scale DNA logic system design. DENA is an FPGA-induced architecture that consists of a two dimensional array of configurable logic blocks [22]. DENA is regular architecture consists of the arbitrary number of DENA Clusters (DC) which can be configured to implement a 4-input logic function. Figure 5 shows the general structure of DENA micro-architecture with 2 * 2 DCs. As shown in this figure, each DC is implemented on a 3 * 3 grid of a microfluidic biochip.

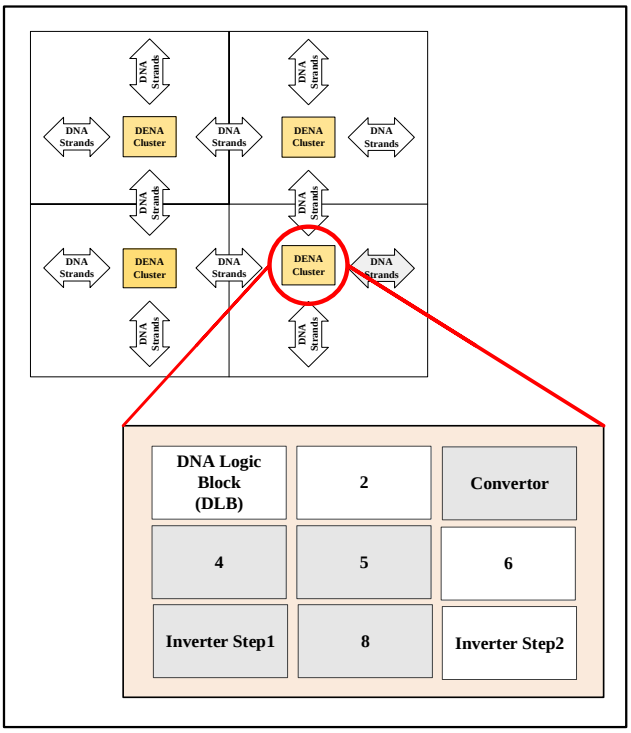

Figure 5. *A simplified view of the DENA architecture proposed in [16].*

Detailed structure and operation of this micro-architecture is described in [16]. We used the DENA [16] as the logical micro-architecture. We made some improvements on DENA to increase its capabilities fault tolerant implementation. At the first of next section these improvements are described. The main contributions of this paper can be defined as follows:

1) To design a buffer gate in modifying of fluctuations in concentrations: each buffer gate will occupy two tiles in microfluidic biochip which can cause additional system overhead, but it is necessary to fix errors. Therefore, the purpose of this paper is to use the minimum number of buffers in order to achieve the minimum error rate.

2) To address occurrence of a fault in microfluidic surface using normal distribution: The probability of occurring a fault in microfluidic surface is investigated using two different approaches. 1- occurrence of a fault over all chip surface using normal distribution model. 2- occurrence of a fault on each electrode of microfluidic using normal distribution model.

Briefly, for implementation of DNA computing, this paper proposed a solution for detecting and fixing of fault in microfluidic biochips

2. MODIFIED MICRO-ARCHITECT-URE

One of the main drawbacks of DNA computing has been historically that chemical reactions between the DNA strands are manual operations and do not seems to be automated easily. However, considerable successful improvements are reported recently on using the DMFBs as the execution platform of DNA reactions. Recently, microfluidic platforms have been used for various steps of the DNA computation. This technology reduces the time of DNA computations and improve controllability and automation of DNA operations [22]. Furthermore, microfluidic biochips can be utilized for providing scalability and flexibility demanded by large-scale DNA logic circuit designs [18]. In fact, microfluidics is the suitable platform for DNA computing. On the other hand, one of the challenges is the creation of errors which has confronted DNA computing with problem. Occurrence of a fault in DNA computing based on microfluidic biochips can be categorized as follow:

1) Increase the probability of occurring a fault in unwilling reactions, which could be due to inadequate system design or incorrect selection of strands. In addition, this probability will rise with an increase in the number of circuit stages.
2) Creation of fault digital microfluidic computing.

Both mentioned cases can lead concentration reduction in output strand and create fault in output logic. To overcome the faults caused by concentration reduction, this paper introduced a modifier buffer. In the following more details on this are presented.

2.1. Modifier Buffer Design

Design of buffer gate has been a serious challenge in DNA logic gate design. We proposed an innovative buffer gate by splitting the buffer operation in two stages (Figure 6). At the first stage, input strand reacts with threshold, gate and fuel strands to generate the output of stage 1, which is then fed into stage 2 as an input. At the second stage, output of the first stage reacts with gate to generate the final output.

Buffer gate improved concentration level without logic exchanged. In fact, if concentration decreases (logic ‘1’) or increases (logic ‘0’) due to the occurrence of error and if this change in concentration is less than 15% of the initial concentration, then the concentration can be returned to initial level using modifier buffer. It is important to note that input strand into buffer and output stand given by the second stage of buffer have the same Nucleotide structure. The structure of proposed buffer is explained in the following.

The input concentration of the first stage is between 750nM to 900nM (as logic ‘1’) or less than 250nM (as logic ‘0’), threshold concentration is 600nM and concentrations of gate1 and gate2 in step1 are 1000nM and 1500nM respectively. In addition, fuel concentration is 2000nM and convertor gate concentration is 1000 nM. These two stages are implemented in 2 distinct microfluidic tiles.

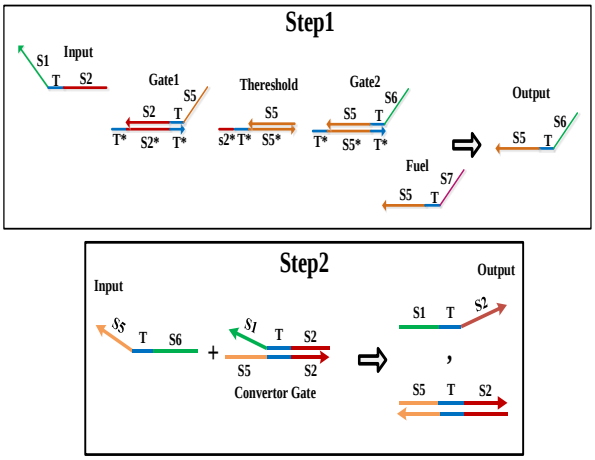

Figure 6. *Internal operations of the proposed buffer.*

As shown in Figure 6, input strand is S1.T.S2, threshold strand includes the upper strand S5 and lower strand s2.T.S5 and gate1 consists of two strands in upper and lower, T.S2.T.S5 and S1.T.S5 respectively. Moreover, strands s2 is sub-strands of S2 and gate2 is Ts5.T.S6. Interactions of internal strands of buffer gate are illustrated in Figures 6-A and 6-B.

These reactions are described as follows.

Step1: Each buffer logic gate has one input, two gate strands, threshold, fuel, representation and output strands that produced after all reactions. Input reacted with gate 1 then produced upper strand S2TS5, this strand quickly consumed with threshold strand (threshold strand consist of two toehold domain), therefore threshold concentration can control S2TS5 concentration. Remaining S2TS5 reacted with Gate 2 and they produce S5TS6 which is output strand; Fuel strand increases the chance of reactions to reach final.

Step2: In this proposed buffer, the structure of input and output strands should be the same, but the structure of output strands of a buffer step1 is completely different from its input structure. Buffer step2 used for change the formation of output strands to the input strands of buffer step1(final output strand is like the first input strand). which is essential in order to avoid changes in architectural structure when using a buffer. The presented convertor structure consists of one input strand (S5TS6) and one gate strand (Convertor Gate). As shown in Figure 6-B, input strand reacted with lower strand of convertor gate to produce a double strand (S5.T.S2) and then output strand is generated which is an upper strand (S1.T.S2). The generated strand is a standard input strand.

These two steps are implemented on 3 microfluidic tiles. Step1 is performed in one tile, then it is latched for one cycle in the next tile and finally step2 is performed in the new next tile.

In contrast to the benefits of using buffer gate, they will occupy three tiles on a digital microfluidic as mentioned. And in addition of space occupying, they will add three delay units to runtime computation as well. To reduce overhead caused by additional buffers, it is necessary to use buffers only when the concentration changes lead error (only to prevent the occurrence of error caused by concentration reduction).

To minimize the number of modifier buffers and thereby reduce the overhead caused by them, it should be noted that there would be no problem if the decrease or increase in concentration does not exceed 15%. Therefore, buffers will only be used in the computational process if the total number of faults exceed 15% at the next stage, in this case, buffers are applied to previous stage of calculation and modify the fault concentration level and consequently fault will not occur in the next stage.

There are two approaches to evaluate the occurrence of fault in system. In the first approach, it is assumed that in each electrode with a normal distribution there will be a fault which leads to decrease or increase of 0% to 15% in output concentration (decrease when generating output in logic ‘1’ and increase when generating output in logic ‘0’), therefore 10% of the total electrodes in chips are in fault. Although this approach is not

accurate enough, it is very easy to evaluate. In the second approach, error can be occurred across the entire surface of microfluidic chip based on two-dimensional normal distribution. In this case, the same as first approach, this fault will be between 0% and 15% of output concentration. In the following, both approaches are explained in more details.

3. FAULT CALCULATION USING FIRST APPROACH

As mentioned before, in this case only 10% of the cells have a fault between 0% and 15%. This fault is appeared by reduction of output concentration in logic '1' and increase of output concentration in logic '0'. To investigate the fault some assumptions are required. These assumptions are mentioned in the following and have extracted statistically in numerous experiments:

1) Maximum 10% of cells have a fault at the surface of microfluidic chip.
2) Error will occur by decreasing or increasing of 1% to 15% of the initial concentration.
3) Reduction of fault which is higher than 15% is rarely happened, but if it occurs then there is no way to compensate.

3.1. Calculation of Minimum Number of Required Buffer in Order to Avoid Creating Modifiable Errors in System

As mentioned previously, with increasing the number of modifier buffers, system overhead will be increased (each modifier buffer consists of two stages which occupy three electrodes in microfluidic chip. On the other hand, if we do not use the modifier buffer then the unmodifiable errors will be occurred by increasing the modifiable errors. As already mentioned above, it is assumed that the cells with compensated error have a concentration reduction from 0% to 15%. After doing measurement, we always have data (numbers) which we attempt to discover the relationship between them or classify them in order to analyze them. To do this, we need to know how data are distributed. Simply, data distribution indicates how spread out or compact our collected data is. Normal distribution(Gaussian) is one of the most important distribution in probability theory.

Normal distribution has two parameters, the mean and the standard deviation. In normal distribution, 68% of data are within one standard deviation of mean and 95% of data are within two standard deviation of mean and 99.7% of data are within three standard deviation of mean. As mentioned earlier, the compensated error range is between 0% and 15%, therefore the mean and standard deviation are 7.5 and 3.75 respectively using the equations 1 to 4.

$$\mu = (\text{Min} + \text{Max})/2 \quad (1)$$
$$\mu = (0+15)/2 = 7.5 \quad (2)$$
$$\sigma = (\text{Max} - \text{Min})/4 \quad (3)$$
$$\sigma = 15/4 = 3.75 \quad (4)$$

where μ and σ indicate mean and standard deviation respectively. We assumed that 95% of errors are modifiable and are within two standard deviation of mean, therefore the standard deviation of 3.75 will be achieved. Bell curve of normal distribution of error is shown in Figure 7.

To achieve the minimum number of required buffers in order to prevent error in system, it should be noted that if the errors are less than the mean, then they can be removed (creating 7.5% concentration reduction) after one step (even with existing of error in the next step) using modifier buffer gate, but errors that are higher than the mean (creating concentration reduction of more than 7.5%) can be modified using buffers in the current step. Thus, according to Figure 8, errors (green dots) that must be modified are 47.5% of the total number of error that occurred.

Since we assumed that 10% of cells are prone to error in each process, therefore the number of required buffer is obtained using equation 5.

10% * n * 47% = total number of required buffer (5)

Where n is the total number of electrodes at microfluidic surface. The experiment showed that most of the errors occurred at the end of the experiment time period on microfluidic surface, consequently it is clear that the buffer cells are more likely to be placed in the last electrodes of experiments. Although investigation of error is easy using the mentioned approach, but it is not accurate enough to calculate the number of required buffer. In the second approach, the number of buffers will be calculated more accurately using computer simulation.

To achieve the minimum number of required buffers in order to prevent error in system, it should be noted that if the errors are less than the mean, then they can be removed (creating 7.5% concentration reduction) after one step (even with existing of error in the next step) using modifier buffer gate, but errors that are higher than the mean (creating concentration reduction of more than 7.5%) can be modified using buffers in the current step. Thus, according to Figure 8, errors (green dots) that must be modified are 47.5% of the total number of error that occurred.

Since we assumed that 10% of cells are prone to error in each process, therefore the number of required buffer is obtained using equation 5.

10% * n * 47% = total number of required buffer (5)

Where n is the total number of electrodes at microfluidic surface. The experiment showed that most of the errors occurred at the end of the experiment time period on microfluidic surface, consequently it is clear that the buffer cells are more likely to be placed in the last electrodes of experiments. Although investigation of error is easy using the mentioned approach, but it is not accurate enough to calculate the number of required buffer. In the second approach, the number of buffers will be calculated more accurately using computer simulation.

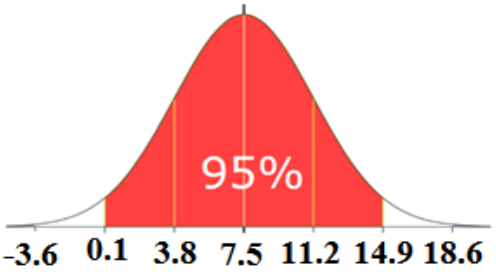

Figure 7. *Normal distribution of concentration reduction rate in modifiable errors.*

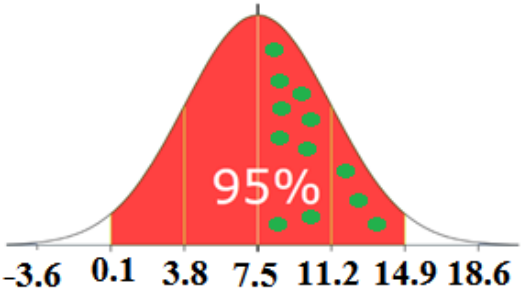

Figure 8. *Green dots shows the errors interval that must be modified by buffers.*

4. ERROR-CHECKING USING SECOND APPROACH

In the second approach, each electrode may have an error in the range of 0% to 15%. This error rate occurs with normal distribution across the entire surface of microfluidic chip. The occurrence of error on the surface of microfluidic chip is shown in Figure 9. The error in each electrode will be calculated by mapping the vertical error graph on the surface of that electrode. As mentioned earlier, in order to perform calculation, it is essential to move the droplets containing DNA strands at level of digital microfluidic. With moving of droplets in level of digital microfluidic, we can use modifier buffer whenever the total number of errors in electrodes reaches 15%.

Simulation of the proposed algorithms was investigated using Matlab (version

R2019a) tools. Simulation, regarding the various possibilities for moving was considered in implementing of each system on surface of digital microfluidic. The maximum number of buffers was used as the number of required buffer in order to avoid modifiable error in each system. The result of modifier buffer simulation and both error-checking approaches are described in the next section.

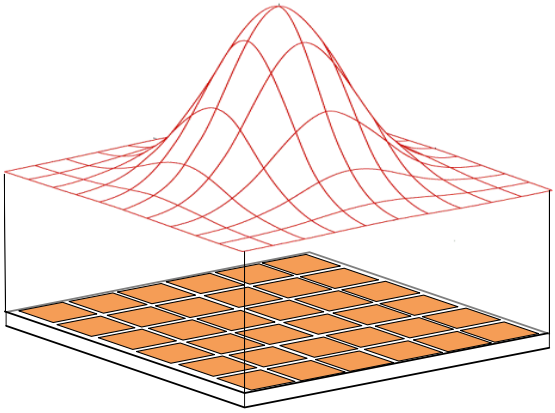

Figure 9. *normal distribution of the occurrence of error on surface of digital microfluidic.*

5. SIMULATION RESULT

As discussed earlier, in this paper a buffer is presented in order to decrease the number of error occurring in DNA computing based on digital microfluidic. Due to overhead caused by buffer (each buffer will occupy two tiles on the surface of microfluidic), we used two approaches based on normal distribution and we will estimate the number of buffers and their location. In the following section the simulation process and design verification are evaluated.

5.1. Buffer Gate Simulation Result

In this paper, we designed a buffer gate to modify the output concentration level. As mentioned previously, in the normal mode when no error is occurred, the output concentration for logic 1 and logic 0 are 900nm and 100nm respectively. Therefore, error means that the output concentration is less than 900nm in logic ‘1’ and more than 100nm in logic ‘0’. The proposed buffer in this paper modifies the concentration of more than 750nm and return it back to more than 900nm and also modifies the concentration of about 150nm and return it back to less than 100nm. The circuit are modeled using the DNA descriptive language and simulated the descriptions using VisualDSD toolbox [23]. VisualDSD is a widely used research tool to simulate the DNA reactions and computes the density of output strands based on the input and internal strands of the circuits. VisualDSD has three possible choices for simulation: stochastic, deterministic, and JIT. In this paper, stochastic modeling was used because the results of stochastic modeling are consistent with real experimental results. Important simulation parameters are shown in Table 1. Simulation was done using a stochastic model and strand concentration listed on this article in related place. Rate of the reactions depends on toehold dissociation and binding rate. This rate is illustrated in Table 2.

Table 1. *Details of Simulation Parameters.*

Parameter	Value
Temperature	25 ˚C
Strand Length	20 (nt)
Strand’s toehold length	4 (nt)
Strand Length	20 (nt)

Figure 10 shows the concentration of buffer gate output strands in step1 that are generated by VisualDSD. It is worth noting that inputs have two different concentrations; 150nM for logics ‘0’ and 750nM for logic ‘1’. As can be seen in this figure, output concentration changes from 0nM to 100nM for logic “0” and from 900nM to 1000nM for logic “1” that makes reasonable noise margin.

Table 3 shows the concentration level of the strands output when input is changed from 0nm to 150nm and from 750nm to 900nm for logic ‘0’ and ‘1’ respectively. As can be seen in this table, level of ‘0’ and ‘1’ outputs have fully acceptable range.

Table 2. *Toehold Reaction Rate.*

Reaction type	Rate	
	Normal	Fast
Toehold dissociation	0.1126	0.95
Toehold binding	3 * 10	1.2 * 10

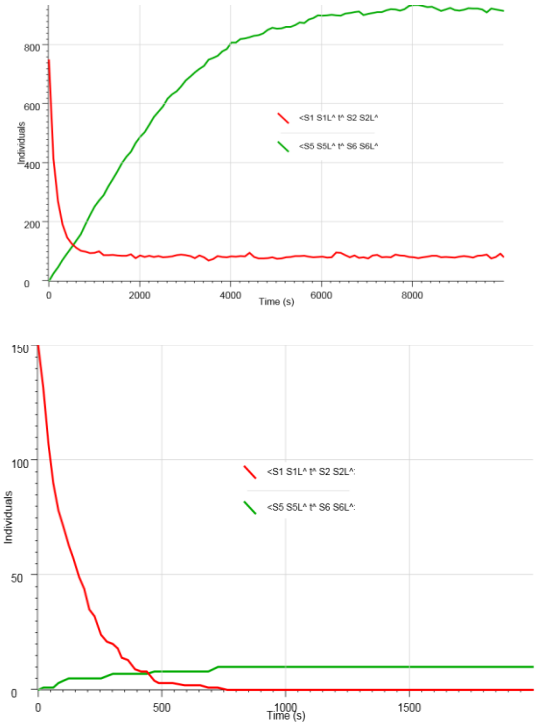

Figure 10. *Simulation result for modify concentration with modifier buffer.*

Table 3 illustrates the output concentration for various range of input concentration. IC and OC show the concentration of Input concentration and output concentration respectively. It is resulted after VisualDSD simulation tool. Finally, column LC show the logical correctness of the buffer output (Logic is correct if LC 'yes'). It is worth to note that high-concentration (>900nM) shows logic '1' and low-concentration (<100nM) represents logic '0' in this table. Table 3 shows that the logical output of the modifier buffer is correct (without error).

5.2. Calculating the Number of Required Buffer Using First Approach

The number of required buffers to modify the output concentration level when an error is occurred are investigated in the remainder of this section.

Table 3. *Input and output concentration for modifier buffer.*

IC	OC	LC
900	1000	Yes
850	997	Yes
800	990	Yes
770	940	Yes
750	914	Yes
700	890	No
600	708	No
400	240	NO
180	130	No
150	98	Yes
100	68	Yes
95	62	Yes

The number of buffers need to be chosen in such a way as to reduce the error and minimizes overhead for the implemented circuit. To do this, as mentioned in the previous section, there are two approaches. In the first approach, the number of required buffers which is based on the number of required tiles for implementation of circuit is calculated using equation 5, and is shown in table 4, where #tiles represents the number of required tiles for implementation of test bench with different complexity, and #buff indicates the number of required buffers. The precise location of these buffers is not predicted and we can only point out that due to the increasing number of errors in the final stages of calculation, it is more reasonable to use buffers at the final stages of calculation. We can also maintain a history of calculation in order to predict the location of error occurring. Then we can use buffers in a location with higher probability of error in regard to the prediction.

Table 4. *number the required buffer in first approach*

Benchmark	Complexity	#IO	#tiles	#buffer
C17	6	7	63	3
C432	160	43	1638	77
C1908	880	58	2646	124
C3540	1669	72	8712	409
C6288	2406	64	16380	770

Table 4 shows the physical implementation of IBM benchmarks on microfluidic biochip. The synthesized circuits are implemented on a Programmable Bio-Cell Matrix (PBCM) architecture [24] using SSS toolbox. As shown in the table 4 Buffers will take a small portion of the chip area (about 4.4%) therefore this overhead can be ignored.

5.3. Checking the Number of Buffers and Their Location Using Second Approach

In the previous section, an approach was presented in order to reduce the modifiable error rates in the computational system implemented on DNA architecture. Modifier buffers lead overhead in terms of occupied space and latency in system. Therefore, decreasing the number of buffers will increase the system performance. On the other hand, the number of modifiable errors will be increased without using buffers. Table 5 shows the number of required buffers in regard to the number of applied electrodes on the surface of digital microfluidics. Table 5 indicates, as the number of applied electrodes on the surface of microfluidics to do calculation increases, the number of modifier buffers also increases. This result is quite expected, since the probability of error occurring follows normal distribution function on the surface of microfluidics chip. Therefore, the probability of error occurring is higher in the central electrodes. Electrodes were placed in different locations of microfluidics chip in order to calculate the number of buffers and the maximum number was considered as the number of required buffers.

Table 5*. Number of required modifier buffers in order to reduce the modifiable error rate.*

Benchmark	Complexity	#IO	#tiles	#buffer
C17	6	7	63	5
C432	160	43	1638	43
C1908	880	58	2646	98
C3540	1669	72	8712	342
C6288	2406	64	16380	590

That shown in table 5 the number of required buffers was significantly lower than the previous approach.

6. CONCLUSION

The main challenge for design of DNA-based logic systems is concentration changes due to error. The concentration changes lead error in the final result of calculation. By increasing the size of implemented system, the probability of unwilling reactions will be increased and also we expect some changes in concentration of final output strands in compare to concentration. In this paper, a new buffer gate proposed that enables cascading of multi-stage logic circuits with increased the system reliability. The proposed buffer will occupy two tiles on the level of digital microfluidic. Therefore, the area will be increased as the number of buffers increases. On the other hand, the reliability will be decreased as the number of buffers decreases. Therefore, it is essential to make a trade-off between the number of buffers and fault tolerantly. With less than 4.4 tiles overhead, reliability can be increased significantly. With The overhead caused by buffer gates will be reduced with optimal use of them. Therefore, this paper used the normal distribution to calculate the minimum number of buffers in a way that does not reduce the reliability.

REFERENCES

1. Currin, A., Korovin, K., Ababi, M., Roper, K., Kell, D. B., Day, P. J., King, R. D., (2017). "Computing Exponentially Faster: Implementing a Non-Deterministic Universal Turing Machine Using DNA", *Journal of the Royal Society Interface*, 14(128).
2. Adleman, L. M., (1994). "Molecular Computation of Solutions to Combinatorial Problems", *Science*, 266(5187): 1021-1024.
3. Sanches, C. A. A., Soma, N.Y., (2016). "A General Resolution of Intractable Problems in Polynomial Time Through DNA Computing", *BioSystems*, 150: 119-131.
4. Zhao, K., Wang, Z., Qin, J., Lu, Y., (2015). "A New Biological DNA Computational Algorithm to Solve the k-Vertex Cover Problem", *Journal of Computational and Theoretical Nanoscience*, 12: 524-526.
5. Seeman, N. C., (2010). "Nanomaterials Based on DNA", *Annual Review of Biochemistry*, 79: 65-87.
6. Livstone, M. S., Landweber, L. F., (2004). "*Mathematical Consideration in the Design of Micro Reactor-Based DNA Computers*", In 9th International Workshop on DNA Computing (DNA9), 180-189.
7. Sakamoto, k., Gouzu, H., Komiya, k., Kiga, D., Yokoyama, S., Yokomori, T., Hagiya, M., (2000). "Molecular Computation by DNA Hairpin Formation", *Science*, 288: 1223-1226.
8. Qian, L., Winfree, E., (2014). "*Parallel and Scalable Computation and Spatial Dynamics with DNA-Based Chemical Reaction Networks on a Surface*", DNA Computing and Molecular Programming, 8727: 114-131.
9. Qian, L., Winfree, E., (2009). "*A simple DNA Gate Motif for Synthesizing Largescale Circuits*", 14th International Workshop on DNA Computing, 70-89.
10. Qian, L., Winfree, E., Bruck, J., (2011). "Neural Network Computation with DNA Strand Displacement Cascades", *Nature*, 475: 368–372.
11. Fan, D., Wang, K., Zhu, J., Xia, Y., Han, Y., Liu, Y., Wang, E., (2015). "DNA Based Visual Majority Logic Gate with One-Vote Veto Function", *Journal of Royal Society of Chemistry*, .6(3): 1973-1978.
12. Hemphill, J., Deiters, A., (2013). "DNA Computation in Mammalian Cells: MicroRNA Logic Operations", *Journal of American Chemical Society*, 135(28): 10512–10518.
13. Wuab, L., Qu, X., (2015). "Cancer Biomarker Detection: Recent Achievements and Challenges", Chemical *Society Reviews*, 44(10): 2963-2997.
14. Grissom, D., Brisk P., (2012). "*Fast Online Synthesis of Generally Programmable Digital Micro-fluidic Biochips*", Proceedings of the 8thIEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, 413-422.
15. Noort, D. V., (2005). "*A Programmable Molecular Computer in Microreactors*", 10th International Workshop on DNA Computing (DNA10), 365-374.
16. Beiki, Z., Jahanian, A., (2017). "*DENA: A Configurable Micro-architecture and Design Flow for Bio-medical DNA-based Logic Design*", IEEE Transactions on Biomedical Circuits and Systems, 11(5): 1077-1086.
17. Amos, M., (1997). "*DNA Computation*", PhD Thesis, University of Warwick, UK.
18. Cannon, B. L., Kellis, D. L., Davis, P. H., Lee, J., Kuang, W., Hughes, W. L., Graugnard , E., Yurke, B., Knowlton, W. B., (2015). "Excitonic AND Logic Gates on DNA Brick Nanobreadboards", *ACS Photonic Journal*, 2(3): 398–404.
19. Yurke, B., Turbereld, A., Mills, A, Simmel, A., Neumann, J., (2000). "A DNA fuelled Molecular Machine Made of DNA", *Nature*, 406: 605–608..
20. Zhang, D.Y., Seelig, G., (2011). "Dynamic DNA Nanothechnology Using Strand-Displacement Reactions", *Nature Chemistry*, 3: 100-113.
21. Abdoli, A., Jahanian, A., (2015). *"Fault-Tolerant Architecture and CAD Algorithm for Field-Programmable Pin-Constrained Digital Microfluidic Biochips*", CSI Symposium on Real-Time and Embedded Systems and Technologies (RTEST), 1-8.
22. Grissom, D., Curtis, D., Windh, S., Phung, C., Kumar, N., Zimmerman, Z., O'Neal, K., McDaniel, J., Liao, N., Brisk, P., (2015). "An Open-source Compiler and PCB Synthesis Tool for Digital Microfluidic Biochips", *Integration The VLSI Journal*, 51:169-193
23. VisualDSD, (2014). [Online]. Available on: http://dsd.azurewebsites. net/beta.
24. Taajobian, M., Jahanian, A., (2016). "Higher flexibility of reconfigurable digital micro/nano fluidic biochips using an FPGA-inspired architecture", *Scientia Iranica*, 23(3): 1554-1562.