Implementation of Reversible Multiplier Circuit Using Deoxyribonucleic Acid

Ankur Sarker[1], Mohd. Istiaq Sharif[2], S. M. Mahbubur Rashid[2] and Hafiz Md. Hasan Babu[1]
Department of Computer Science and Engineering[1], Department of Genetic Engineering and Biotechnology[2]
University of Dhaka, Dhaka- 1000, Bangladesh.
E-mail: ankur.sum@gmail.com, istiaq.sharif@yahoo.com, suzandu@gmail.com and hafizbabu@hotmail.com.

Abstract

In this paper, we realize the reversible multiplier circuit using Deoxyribonucleic Acid (DNA). Due to reversible logic's emerging characteristics, it has drawn great attention in recent years. As multiplication operation consists of several shift and addition operations, we use shifter and adder circuits as building blocks to construct multiplication circuit. We also present an algorithm for depicting overall procedures of multiplication operation using an example. The proposed circuit is faster, required less space and power due to parallelism, replication properties, compactness and formation of DNA strands, respectively. Additionally, the run time complexity of our proposed system is $O(m)$ *instead of* $O(m(\ln_2 n)^2)$ *in existing DNA-based system, m and n are the bit length of multiplier and multiplicand. Also, proposed system needs* $u+3.2^n$ *DNA signals while the existing system needs* $u.2^n$*, u is the extra tag.*

***Index Terms**: Reversible logic, DNA computing, Multiplication, DNA-based multiplier circuit.*

I. Introduction

Since 1994 Adleman has solved the *HPP* (Hamilton Path Problem) with 7-vertices using DNA [1]. In 1995, Kari and Paun proposed the Sticking System to solve the binary information where two types of strands are used for the binary representation: *single-stranded DNA* and *double-stranded DNA* for 0 and 1, respectively [2]. The information in DNA is stored as a code made up of four chemical bases: *Adenine* (*A*), *Guanine*(*G*), *Cytosine*(*C*) and *Thymine*(*T*). DNA bases pair up with each other, *A* with *T* and *C* with *G*, to form units called base pairs [3].

Reversible logic was introduced with a view to minimize the energy loss of a circuit. According to Landauer, in irreversible circuits, every bit of information loss generates *KTln2* Joules of energy where *k* is the Boltzmann constant of 1.38×10^{-23} *J/K* and *T* is the operating temperature [4]. According to Bennet, zero energy dissipation would be possible only if the network consists of reversible gates [5]. By preserving reversibility, it ensures that the number of input vector is equal to the number of output vector, so that it reduces the information loss.

Followings are the main advantages of using DNA-based circuits over the existing silicon chip:

- In double strands DNA, the data density will be one base per square nanometer and the data density will be over one million Gbits per square inch where in typical high performance hard drive, the data density is about 7 Gbits per square inch [3], [6], [7].
- Base pair complement gives a unique error correction mechanism which works as like RAID 1 array [8].
- As many copies of the enzyme can work on many DNA molecules simultaneously. It can work in a massively parallel fashion [9].
- In DNA replication, enzymes start on the second replicated strand of DNA even before they are finished copying the first one. So, data rate jumps to 2 times of initial speed (initially it is 1000 bits/sec). After each replication is finished, the number of DNA strands increases exponentially. Suppose, after 30 iterations, it increases to 1000 Gbits/sec [10].
- DNA is a stable molecule, never suffers any changes (mutation) unless it faces harsh environment [11].
- DNA logic gates can be preserved for a very long time by maintaining and varying the temperature [12].
- A tiny energy is required to break the bound when operating DNAs. For example, the energy required to break the bond between *A* and *T* is ≈21 KJ/mol where *A* denotes adenine and *T* denotes thymine. The same 21 KJ/mol will be gained if a bond between them is formed again. That means energy is reserved [13].

So, any DNA-based composite circuit requires less space, and it has self error recovery capability, parallelism and faster read-write capability over any kind of existing circuits.

II. Basic Definitions and Properties

In this section, we present the basic definitions and properties which are related to reversible logic and DNA.

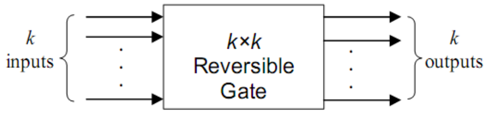


Fig. 2.1: *k*×*k* Revesible Gate.

Let the input vector be I_V and output vector be O_V where $I_V = (I_1, I_2, ..., I_k)$, $O_V = (O_1, O_2, ..., O_k)$ and $I_V \leftrightarrow O_V$. A ***k×k reversible gate*** is a *k*-input and *k*-output circuit that produces a unique input-output mapping [5]. For example, a *3×3 Tofolli* gate is a *3×3* reversible gate. Unused outputs of a reversible gate are known as ***garbage outputs***, are only used to maintain the reversibility.

DNA is the deoxyribonucleic acid. Two chains of DNA are in a double helix. The strands are joined by hydrogen bonding between the bases of opposite strands, to form the base pairs [3]. ***DNA denaturation*** also known as DNA melting, is the process by which *dsDNA* (double-stranded) unwinds and separates into single strands through the breaking of *H-bonds* between the bases and becomes *ssDNA* (single-stranded). Both terms are used to refer to the process as it occurs when a mixture is heated, although "*DNA*

978-1-4799-3163-7/13/$31.00 ©2013 IEEE

Denaturation" can also be referred to the separation of DNA strands induced by chemicals. Again, "***DNA renaturation***" means the formation of *dsDNA* (double-stranded) from *ssDNA* (single stranded) with the help of base pairing by using H-bonds to make a complementary sequence. This term is often used to describe the reformation of complementary strands that were previously separated by heat (thermally denatured) or by use of some chemicals (chemically denatured). In final step of *DNA renaturation,* annealing, also known as hybridization, occurs by complementary bases formed using *H-bonds*. The *5′-end* designates the end of the DNA or RNA strand that has the fifth carbon of the deoxyribose or ribose at its terminus where a phosphate group is attached. The *3′-end* of a strand the another end of the DNA or RNA strand and is so named due to its terminating at the hydroxyl group of the third carbon in the sugar ring.

III. Design of DNA-Based Reversible Multiplication Circuit

In this section, we briefly describe our proposed DNA-based reversible multiplication operation which consists of DNA-based reversible shifter and adder circuit.

3.1. Multiplication

A Multiplication operation, works on two numbers *Multiplicand* and *Multiplier,* produces final result- *Product*. It usually uses each single bit of *Multiplier* to shift the value of *Multiplicand* and add shifted result with partial *Product*. For an *n*-bit *Multiplier*, this process repeats *n* times; finally, *Product* is produced. Our proposed multiplication circuit consists of three steps: working on single bit of Multiplier, shifting the Multiplicand and adding shifted multiplicand with partial product to produce final *Product*. If *Multiplier* is *n* bits and *Multiplicand* is *m* bits*,* then *Product* is *(n+m)* bits.

3.2. DNA-Based Shifter Circuit

To construct the shifter circuit, we adopt the procedure and technical parameters (with slight modifications) of prior work [15]. Two single stranded DNA (*ssDNA*) molecules are used as control inputs and another one is used as target input. The final output is based on two control input signals, it is generated when appropriate pre-designed *ssDNA* segments anneal bind with the target input and make it double stranded (*dsDNA*). This double strand is denatured for separation of the smaller strand by capillary electrophoresis. Fig. 3.1 shows these overall procedures. This smaller strand is ready to be annealed with the modified input strand (having *poly-A* tail and inosines, I, binds with all the bases and indicates no value after binding with the bases, it used here for blocking one bit). It is important to note that this is the first point of difference between left shifter and right shifter function mechanisms; for right shifter *poly-A* tail will be at 5′ end, while for left shifter tail will be at 3′ end. Meanwhile, all others are transferred to storage for reuse (at upper right side of Fig. 3.1). This double strand is now ligated with another input (two base pairs long double strand DNA fragment that adds either 1 or 0 bit signal) on the opposite site of flanking *poly-A* nucleotide tail. This ligated double strand is the output for the first round of shifter function and procedures are continued for *n* times for n-bit shifting, [see in Fig. 3.1]. Two types of shift could be possible- one is shift with rotation, another is shift without rotation. First, operand and input strand are annealed in a compartment. This double strand is denatured for separation of the smaller strand by capillary electrophoresis.

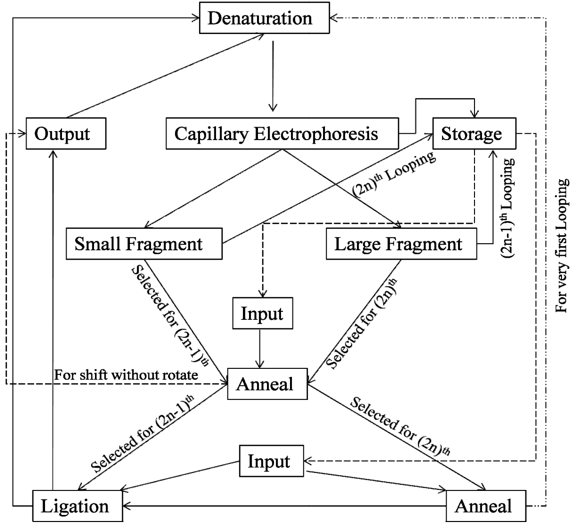


Fig 3.1: The overall essential steps of DNA-based reversible shifter function (with rotation) applicable to any calculation [16].

This smaller strand is ready to be annealed with the modified input strand (having inosines for blocking one bit), when all other things like larger strand is transferred to storage. This double strand is now ligated with another input (two base pairs long double strand) on the opposite site of flanking poly nucleotide tail. This ligated double strand is the output of first loop. Second, the double stranded output of first loop is denatured for separation of larger strand by capillary electrophoresis. This larger strand is ready to be annealed with the meaningless shortest input strand [*MSIS*, (5'-II' UUTTT)], when all other things like smaller strand is transferred to storage. The larger strand is now covered by *MSIS* partially and ready to be annealed again with *MSOS* [meaningful (one bit) shorter operand strand] on the rest larger single stranded portion. This double strand is now ligated with another input (two base pairs long double strand) on the opposite site of flanking poly/mono nucleotide tail.

This ligated double strand is the output of second loop. This double strand is denatured to be separated into large, medium and small strands where the medium one is the operand strand (same as initial operand) and the large one is the input for third loop. Now, 3rd, 5th…(2n+1)th loop will follow the same path as the first loop. And 4th, 6th…(2n)th loop will follow the same path as the second loop [16]. In case of shifter without rotate (SnR), all other steps are needed except the ligation; and with progression of steps there will be a consecutive increase of inosine base in the modified input signal.

3.3. Procedure of DNA-Based Multiplication

The proposed multiplication operation uses previously described shift operation and add the partial results using existing method [15]. Fig. 3.2 shows the operations of multiplication. The DNA strand representing Multiplicand (*Mt= Ot1*) will pass through the left shift without rotation *{[L]SnR}* operation to generate output *Ot2*, *Ot3*,…,*Otp* after *p-1* looping. Here, *p* is the number of bits present in the Multiplier, *Mr*. The upper left side of Fig. 3.2 shows the operations on Multiplicand, *Mt*. Again, the DNA strand representing Multiplier, *Mr* will pass through the right shift with rotation *{[R]SR}* operation to generate output *Or1, Or2,…,Orp* after *p* looping. Upper right side of Fig. 3.2 also shows the operations on Multiplier, *Mr*. In both type of shifter operation, after initial input, *Mt/Mr,* the input for one turn will be the output of immediately previous turn.

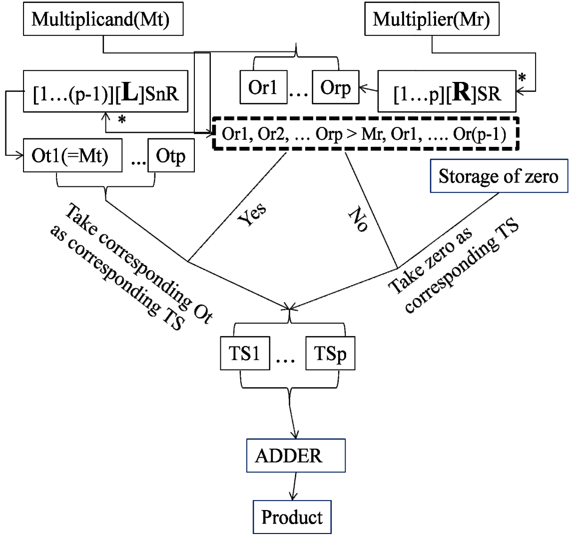


Fig. 3.2: Procedures of DNA-based reversible multiplication.

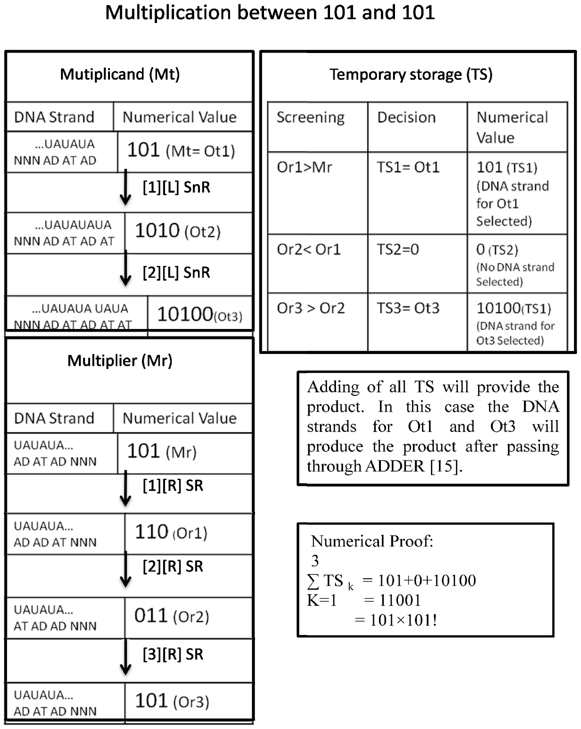
Now after screening if, Or_1 is larger than *Mr* then DNA strand representing Ot1 will be transferred to temporary storage, *TS* and taken as *TS1*. And, if *Or1* is smaller than *Mr,* then no DNA strand will be transferred to *TS* and *TS1*=0, (which is shown using the dashed outline box in Fig. 3.2). Similarly, if *Or2* is larger than *Or1*, then *TS2=Ot2*. And, if *Or2* is smaller than *Or1*, then *TS2*=0.

So, the simplification of above: every output of *[R]SR* will be compared with its input. If the output is larger, then corresponding output of *[L]SnR* will be taken as temporary storage. On the other hand, if the output is smaller, then *TS* will be null. Mathematically if *Orn*>*Or*(n-1), then *TSn*=*Otn*. And if *Orn*<*Or*(n-1), then *TSn*=0. Another case, if the output and input of *[R]SR* is equal, then the *Mr* is definitely all 1s or all 0s. After screening, if *Mr* contains 1, then every *Ot* will taken as every corresponding TS and, if *Mr* contains 0, then all *TS* will be 0. Finally, all contents of temporary storage will be passed through adder circuit and produce the *Product*. Mathematically, $\sum TS = Product$. So, addition between partial result and *TS* (Temporary Storage) is applied. After n times, it produces the overall result of multiplication operation, *Product*. Here, *TS1*, *TS2*…*TSp*; it means all *TS* are passed through the Adder to get product as output of Adder. So, $\sum_1^p TS$ is equal to final *Product*. Fig. 3.2 shows the procedures of DNA-based multiplier circuit where *Ot*= *Output*(*Mt*), *Or*= *Output*(*Mr*), *TS*= *Temporary Storage*, *p*= Number of bits in *Mr* = Number of loops during executing *SR/SnR*. After initial *Mr* or *Mt*, inputs will be *Or1*, *Or2*,…, *Orp* or *Ot1*, *Ot2*,…, *Otp*, consecutively and *Mt* = *Ot1*.

IV. An Algorithm for proposed DNA-Based Reversible Multiplication

Initialize: Represent Multiplicand as *Mt* (should not be zero) and Multiplier (should not zero) as *Mr*. *Ot* is equal to Output, Mt; *Or* is equal to Output, *Mr*; *TS* is the Temporary Storage and *PMr* is another Temporary Storage. *p* is equal to number of bits in *Mr*, is also equal to Number of loops during executing *SR*/*SnR*. Here, sizes of Mt, TS are equal to *2p*.

```
Step 1: Repeat Step 2 to Step 11 in total of p times
Step 2:   PMr ← Mr;
Step 3:   Apply (Right) Shift with Rotation of Mr;
Step 4:   Or ← output (Mr);
Step 5:   Apply (Left) Shift with not Rotation of Mt;
Step 6:   Ot ← output (Mt);
Step 7:   If Or is greater than PMr
Step 8:             Then do TS ← Ot;
Step 9:   Else do
Step 10:            TS ← Shorage of zero;
Step 11:  Product ← Addition (Product, TS);
Step 12: Product is the final result of Multiplication;
```



Multiplication between 101 and 101

Mutiplicand (Mt)

DNA Strand	Numerical Value
…UAUAUA NNN AD AT AD	101 (Mt= Ot1)
	[1][L] SnR
…UAUAUAUA NNN AD AT AD AT	1010 (Ot2)
	[2][L] SnR
…UAUAUA UAUA NNN AD AT AD AT AT	10100(Ot3)

Multiplier (Mr)

DNA Strand	Numerical Value
UAUAUA… AD AT AD NNN	101 (Mr)
	[1][R] SR
UAUAUA… AD AD AT NNN	110 (Or1)
	[2][R] SR
UAUAUA… AT AD AD NNN	011 (Or2)
	[3][R] SR
UAUAUA… AD AT AD NNN	101 (Or3)

Temporary storage (TS)

Screening	Decision	Numerical Value
Or1>Mr	TS1= Ot1	101 (TS1) (DNA strand for Ot1 Selected)
Or2< Or1	TS2=0	0 (TS2) (No DNA strand Selected)
Or3 > Or2	TS3= Ot3	10100(TS1) (DNA strand for Ot3 Selected)

Adding of all TS will provide the product. In this case the DNA strands for Ot1 and Ot3 will produce the product after passing through ADDER [15].

Numerical Proof:

$$\sum_{K=1}^{3} TS_k = 101+0+10100 = 11001 = 101\times101!$$

Fig. 4.1: Example of DNA-based reversible multiplication operation.

Above algorithm depicts the overall procedures where If *Or* is equals to previous value of *Or* then *Mr* will be screened for 0 or 1. Again, if *Mr* is equal to 1, then all *Ot* will be taken and otherwise *Mr* will be zero and *Product* will be zero. The run time complexity of this algorithm is *O(m)* if m is total bit of multiplier.

Example 4.1: Fig. 4.1 shows an example of our proposed DNA-based multiplication operation with three data paths: *Multiplicand*, *Multiplier* and *Temporary Storage*. For adding all *TS*s, the DNA strands for *Ot1* and *Ot3* will produce the product after passing through the adder circuit [15]. Here, 101 $\binom{\text{UAUAUA}}{\text{NNNADATAD}}$ and 101$\binom{\text{UAUAUA}}{\text{ADATADNNN}}$ are multiplied together, it produces final output 11001 $\binom{\text{UAUAUAUAUA}}{\text{ADADATATAD}}$ where $\sum_{k=1}^{3} TS_k$=101+0+10100=11001 = 101×101.

V. Properties and Comparisons of DNA-based Multiplier Circuit

Theorem 5.1: It is enough to determine positional value in binary by shifting the positional value with rotation.

Proof: Binary number only contains 0, 1; it means right shift any number with rotation produced a number less than the original one if *LSB* contains zero and vice-versa. It completes the proof. □

Theorem 5.2: The proposed DNA-based reversible multiplication circuit performs correctly.

Proof: Our proposed multiplier circuit consists of shifter and adder circuits which are logically reversible and uses DNA bases for representing input and output signals. Adder and shifter circuit perform successfully and produced valid results. By combining those two produces multiplier where it operates using bit by bit value of multiplicand and multiplier. So, our design is flawless as well as proposed circuit. □

Above theorems describes some properties of our proposed design methodology. We also compare our proposed design with existing DNA-based computation systems in Table I.

Table I: Comparisons between the Existing DNA-Based System and the Proposed System.

Parameters	Existing System [17]	Proposed System
Constraint	$u+2n \leq 20$	None
Required DNA signals	$u.2^n$	$u+3.2^n$
Process of DNA formation	Both the denaturing and renaturing are required	None, as the signal is already in renatured form
Probability of hydrolysis	Probability is high	Probability is low
Signal types	Uniform for all operations	Varied according to logical operations
Complexity of DNA formation	Less DNA bases are used, simple	Few more DNA bases are used, complex
Parameters	**Existing System [18]**	**Proposed System**
Number of biological operations	5 operations are used	3 operations are used
Run time complexity	$O(m(ln_2n)^2)$	$O(m)$

Here, $\boldsymbol{m}$, $\boldsymbol{n}$ are the number of bits of multiplier and multiplicand, respectively, and $\boldsymbol{u}$ is the size of extra tags.

From Table I, we can easily conclude despite of having few flaws (including extra tags, generalization process) in our proposed system, it will perform better than other existing DNA-based designs [17], [18].

VI. Conclusions

We have constructed multiplier circuit using deoxyribonucleic acid (DNA) signals instead of using silicon chips, in this study. Our proposed multiplier circuits works into three steps: working on single bit of multiplier, shifting the multiplicand and adding the partial product to produce the final result. To implement DNA signals, we have used some natural and non-natural (*X*, *Ψ* and *D*) DNA bases for providing broaden window of complementary design. We have also presented an algorithm to produce a compact DNA-based multiplier circuit. The run time complexity of proposed method is *O(m),* whereas the run time complexity of the existing method is $O(m(ln_2n)^2)$ [18]. In addition, the use of DNA signals makes our circuit faster [14] and it will be reusable if temperature will be maintained in between a certain range [2].

References

[1] Adleman, Leonard M. "Molecular computation of solutions to combinatorial problems." SCIENCE-NEW YORK THEN WASHINGTON, 1994, pp. 1021-1021.

[2] Kari, L., Paun, G., Rozenberg, G. et al., "DNA computing, sticker systems, and universality", Acta Inform., 1998, 35: 401-420.

[3] J. D. Watson, F. H. C. Crick, "Molecular structure of nucleic acids. A Structure for Deoxyribose Nucleic Acid", The Journal of the American Medical Association, 1993; 269(15):1966-1967.

[4] Keyes, Robert W., and Rolf Landauer. "Minimal energy dissipation in logic." IBM Journal of Research and Development 14, v. 2, 1970: 152-157.

[5] C. H. Bennet, "Logical reversibility of computation", IBM J. Research and Development, vol. 17, no. 6, pp. 525-532, 1973.

[6] Will Ryu, "DNA Computing: A Primer", Ars Technica, 2002.

[7] Yang Yingwei, "DNA COMPUTING: DNA Computers VS. Conventional Electronic Computers", 2002.

[8] H Echols, and M F Goodman, "Fidelity Mechanisms in DNA Replication, Annual Review of Biochemistry", 1999, Vol. 60: 477-511.

[9] Karl, Lila. "DNA computing: arrival of biological mathematics." The mathematical intelligencer 19, v. 2, 1997, pp. 9-22.

[10] Hiasa, H. & Marians, K.J., "Initiation of bidirectional replication at the chromosomal origin is directed by the interaction between helicase and primase." *J. Biol. Chem.* 274, 1999, 27244–27248.

[11] Hauke Clausen Schaumann, Matthias Rief, Carolin Tolksdorf and Hermann E. Gaub, "Mechanical Stability of Single DNA Molecule, Biophysical Journal", Volume 78, Issue 4, 1997-2007, 1 April 2000.

[12] S. Lund and J. Dissing, "Surprising stability of DNA in stains at extreme humidity and temperature"; International Congress Series, Volume 1261, April 2004, Pages 616-618.

[13] Katharina Wendler, Jens Thar, Stefan Zahn and Barbara Kirchner, "Estimating the Hydrogen Bond Energy", *J. Phys. Chem. A*, 2010, *114* (35), pp 9529–9536.

[14] C. Pandu Rangan, V. Raman, R. Ramanujam (Eds.), "How to Compute with DNA", FSTTCS'99, LNCS 1738, pp. 269-282, 1999.

[15] Sarker A., Ahmed T., Rashid S.M. M., Anwar S., Babu H.M.H., "Realization of reversible logic in DNA Computing". In Proceedings of the 11th IEEE International Conference on Bioinformatics and Bioengineering, Taichung, Taiwan, October 2011, pp. 261-265.

[16] Tanvir A., Ankur S., Mohd. Istiaq S., S.M. Mahbubur Rashid, H. M. H. Babu, "A NOVEL APPROACH TO DESIGN A REVERSIBLE SHIFTER CIRCUIT USING DNA" 26th IEEE International System-on-Chip Conference, Germany, September 04-06, 2013, (in press).

[17] V. Gupta, S. Parthasarathy and M. J. Zaki, 1997, "Arithmetic and logic operations with DNA". In Proceedings of the 3rd DIMACS Workshop on DNA Based Computers, Philadelphia, pp. 212-220.

[18] de Santis F., Iaccarino G., "A DNA Arithmetic Logic Unit", WSEAS Transactions on Biology and Biomedicine, vol. 1, October 2004, pp. 436–440.