CHAPTER-6

DNA Based Computing Devices
In the preceding chapters we have seen how silicon based computing devices have been designed from silicon material and molecule based computing devices designed from organic materials. Now in this chapter it is extending the designing and computing principles of biomolecules based electronic devices in specific to DNA (Deoxyribo Nucleic Acid) molecule.

To solve mathematical problem we rush to computers—the awesome electronic machines that have taken over several mechanical and mundane tasks. But when we are thinking about a nature’s liquid computers, right since life began, to perform all the complex functions of a life form. This computer par excellence is a biomolecule—the DNA (Deoxiribose Nucleic acid).

DNA can do all that a computer does. It can store information, compute information and reproduce information just like a computer. Inside a cell, it interacts with the environment and manipulates the signals for its benefit. Most importantly, it can evolve just like an intelligent computer.

DNA computing is new computation paradigms, which uses DNA, biochemistry and molecular biology, instead of the traditional silicon-based computer technologies. DNA computing is interested in applying computer science methods and models to understand such biological phenomena and gain interest into early molecular evolution and origin of biological information processing. The primary advantage of DNA based computation is the ability to handle millions of operations in parallel. DNA computing is fundamentally similar to parallel computing in that it takes advantage of the many different molecules of DNA to try many different possibilities at once[193].
A typical DNA computer processes can be shown like

6.1 Basic Initial Computational approach to DNA Molecule based Computer

Various researchers have been working on to design a DNA computer and its underlying computing aspects. Some of the pioneer and advanced work has been discussed here. Major progress in the application of nucleotides for information processing came about two decades later with Adelman's insight that random oligonucleotides could be the basic tokens for information processing [194]

6.1.1 M. Adleman DNA Computing concept

Leonard M. Adleman of the University of South California, first to pioneer in designing a DNA computer by taking DNA polymerase. He proposed an algorithm for solving Hamiltonian Path Problem using molecular operations [195]. Adleman's experiment ushered in a new computational paradigm for several reasons. First it showed that it is indeed possible to orchestrate individual molecules to perform computational tasks. Second, it showed the enormous potential of DNA molecules for
solving problems beyond the reach of conventional computers that have been or may be developed in the future based on solid-state electronics [196].

Since Adleman’s pioneering experiment, several authors attempted to present efficient DNA algorithms to solve hard problems [198] and simulating conventional computing models such as Turing machines, Finite state automata and splicing system [194].

It reads one strand of DNA strand and makes the other strand according to Watson and Crick model [61]. If one sequence is 5'-GACTACG-3', under DNA polymerase, it makes a new molecule with a sequence 3'-CTGATGC-5'. Adelman’s computing system based on the travelling sales man problem and related to “Hamiltonian path problem” [195] and solved in a test tube using standard method of biology. Moreover Adelman’s design DNA computer in a restricted Model.

Adelman employed enzymes only to stabilize (through covalent bonds) the products of a self-assembly process (hybridization of partially complementary oligonucleotides), but not in the information processing itself, and accordingly did not require enzymes with sequence specificity.

6.1.2 Liptons’ Approach

Lipton [199] presented on early proposal for Boolean Circuit evaluation as a solution to SAT (Boolean formula satisfiability)[197]. Lipton showed how molecular computers could be applied to NP-complete problem and can solve any difficult problem. The underlying principle was the same as Adleman’s approach: generate all possible solutions to the problem, and then gradually filter out strands until any remaining strands must be a solution to the problem. Lipton proposed solving an instance of SAT by starting with a tube containing strands representing all possible assignments to its variables. Lipton proposed extending Adleman’s algorithm ‘in a way that allows biological computers to potentially radically change the way we do all computations, not just HPPs [194].
Lipton’s main contribution lay in his method for encoding arbitrary binary strings as strands of DNA. The approach would be fairly cheap and quick to specify and order them to be individually synthesized for small number of strands. However, it is becoming infeasible if the number of variables (strands) grow [200]. One of the characteristics of the NP-complete problems is that for a small increase in the problem size i.e., the number of variables, the number of possible solutions rises exponentially. For example solving a problem with ten variables, you may need to order $2^{10} = 1,024$ individual strands [200].

Lipton found a way of encoding an exponential-sized pool of starting strands, using only a polynomial number of ‘building-block’ strands. He wanted to build a large number of assignment strands, using a small number of ‘variable’ strands. His key insight was that an arbitrary binary string could be represented as a path through a graph, where each node represented one particular bit (or variable) [200].

Lipton and Adleman used exhaustive search to implement their algorithm. In exhaustive search, all the possible solutions are encoded by strands. Then the solution can be obtained from the exponentially sized initial set, by applying DNA operations. This approach is possible because of the inherent criteria of DNA strands as computing devices [201].

In both Adleman’s HPP and Lipton’s SAT algorithms; the total volume of DNA present in a test tube at any time of the computation grows exponentially as a function of input size. As a result, their algorithms can handle instance size of up to 70 which are within the reach of silicon based computers. Therefore it is urgent to find applications where DNA computers outperform silicon based computers.
6.2 Basic Needs for DNA based logic gate

Referring to Fig.3.2.3. (a) and Fig.3.2.3. (b) of chapter 3 which has shown the nucleotide and double helical structure of DNA respectively, the same structure is modeled for the molecular structure of DNA shown below. DNA can be single strand or double strand. The below figure shows a double strand DNA. DNA is a molecule with linear sequences of nucleotide.

![Molecular structure of DNA](Source: sciencegateway.org)

A prominent difference between solid-state materials and macromolecular materials is the large range of properties found in molecules. Biomolecules are mainly composed from only six (C, H, O, N, S, P) out of the 91 naturally occurring chemical elements. The number of possible compounds that could in principle be formed from these six atoms is very large. Macromolecules occurring in organisms are typically formed from a set of building block molecules. These building blocks link through covalent bonds originating at specific atoms, but can be combined in arbitrary order. The twenty commonly occurring amino acids form such a set of building blocks. Linear polymers form up to a few hundred of these amino acids linked in arbitrary sequential order, constitute an important class of
biomacromolecules, the proteins. Another set of building blocks found in nature are the nucleotides which combine, again in arbitrary order, to long nucleic acid molecules. Two phenomena are key to the interaction and function of macromolecules: self-assembly and conformational dynamics [202]. Both play also an important role for molecular information processing in nature and each serves as a paradigm for man-made molecular computing schemes.

In a DNA molecule, the nitrogen base classify on the basis of their ring structure as: Purine and Pyrimidine. Purine includes Adenine nucleotide (A) and Guanine nucleotide (G). Similarly the Pyrimidine contains Cytosine Nucleotide(C) and Thymine Nucleotide (T).

A nucleotide consists of a 5-carbon sugar (deoxyribose), a nitrogen containing base attached to the sugar, and a phosphate group. The first letter of the bases gives the name to the nucleotides. There are four bases and therefore four nucleotides: adenine (A), guanine (G), cytosine (C) and thymine (T). The carbon atoms of the deoxyribose are numbered 1’, 2’, 3’, 4’, and 5’. The hydroxyl groups on the 5’ and 3’ carbons link to the phosphate groups to form the DNA backbone [4]. One DNA strand is an alternating backbone of deoxyribose and phosphodieser groups that has a polarity (direction) defined from 5’ to 3’.

![Fig. 6.2(b) 5' and 3' based on phosphate and hydroxyl group](image-url)
As discussed in chapter 3 about oligonucleotide, it is a short chain of nucleic acid, a set of nucleotides joined in a single strand. These chains are able to anneal with the complementary sequence of nucleotides. There are several reasons to choose oligonucleotides as inputs and output of logic gates. The advantages are the stability and intracellular structure of the oligonucleotides and specific oligonucleotide can able to produce [203].

DNA molecules can act as elementary logic gates analogous to the silicon-based gates of ordinary computers. Short strands of DNA serve as the gates’ inputs and outputs.

The inputs and output sequence are shown below Fig. 6.2(c). The specific sequence of nucleotides which is the complementary sequence of a certain part of the deoxyribozyme can act as inputs. The output is a cleaved product oligonucleotide resulting from the cleaving of an initial substrate.

![Fig.6.2 (c) Input and Output Sequence of oligonucleotides](image)

6.3 Operation and Methods of designing DNA Computing Device

There are multiple methods for building a computing device based on DNA, each with its own advantages and disadvantages. Most of these build the basic logic gates (AND, OR, NOT, XOR etc.) associated with digital logic from a DNA basis. Some of the different bases include DNAzymes, deoxyoligonucleotides, enzymes, DNA tiling, and polymerase chain reaction.

The inputs can be formed as example $x_1=\text{5'}-\text{GGGGATTAACC-3'}$, $x_2=\text{5'}-\text{GGGAATGTCCC-3'}$, $x_3=\text{5'}-\text{GGGCAGCAGCCC-3'}$, etc. The inputs $x_1$ and $x_2$ can go through hybridization properties to give a specified gate $P$. 
The strands of the DNA as inputs for the gate formation. The output results with several chemical compositions like **hybridization**, **enzyme attachment**, **ligase** etc., and form a complimentary sequence.

**Catalysts** are chemical entities that play a vital role in chemical reactions. A catalyst is recovered after the complete reaction has taken place. In other words, catalysts permit other components to react between them, but do not change themselves. The **enzymes** are catalyst, present in the reactions that involved DNA. Until now most of the proposed models in DNA computing were based in the well known action of the enzymes over the DNA strands. But a new model designed and proposed by Maria Belen Canadas Ruiz-Perez [203] is explained which is based in a new catalyst: catalytic DNA or deoxyribozyme. The intervention of an enzyme is necessary to provoke reactions as normal structure of DNA cannot react alone. For example, to clone DNA the action of the enzyme called helicase to melt the double strand chain is necessary. But DNA can develop catalytic functions by called as deoxyribozyme. The deoxyribozymes are special structures of oligonucleotides configured in such a way that in contact with a fixed substrate, they catalyze a chemical reaction. The deoxyribozyme can produce in lab from RNA (RiboNucleic Acid) which is a ribozyme [204]. A ribozyme generally behaves like an enzyme present in the alive organisms [203]. Different kinds of deoxyribozyme were created like self-cleaving DNA which uses amino acids as cofactors, catalytic DNA of the reactions for self-cloning, DNA that catalyzes its own destruction or the destruction of other sequences of DNA.
The deoxyribozymes used as logic gates in this experiment are called E6 [204] and 8-17 [205, 206]. The E6 deoxyribozyme has a catalytic core and an internal loop (Fig. 6.3(b)). The internal loop can be substituted by a desired sequence; this property will be used to design NOT gates. On the other hand, the 8-17 deoxyribozyme is formed by a catalytic core and a fixed internal loop (Fig. 6.3(c)) that is not possible to change. Stem-loops can be placed in the arms of these oligonucleotides to show annealing properties. When complimentary sequence is not mixed with deoxyribozyme, it is inactive and loop get close. But when the complementary sequence exists in the solution the loop opens and the oligonucleotides anneal and are in active form [203].

The deoxyribozymes used to build up the gates are supposed to be catalyst and must remain invariable after the reaction. For input “0”, these catalytic oligonucleotides do not change and if the input is “1” they become in active form; in other words, the input oligonucleotide is annealed to the deoxyribozyme. The reset of the gate and the removal of the input is made by washing, adding to the solution the complementary sequence of nucleotides of the inputs. The enzymes are limited in diversity but deoxyribozymes have a bigger range of application.

Many recombinant DNA operations use hybridization and are specific to a DNA segment with a prescribed n-mer subsequence. Short strands of single DNA of
length n are called as n-mers. Such recombinant DNA operations include cleavage, separation, detection and fluorescent tagging of specific DNA words. In addition there are specific operations like ligation of DNA segments to form covalent bonds that join covalent bonds together and merging of test tube contents.

The cleavage as shown Fig. 6.3 (d) is the splitting of a molecule into simpler molecules by breaking a chemical bond. In the logic gates, the cleavage is made to a substrate (ended by a fluorescein donor in the 5'-end (F) and a tetramethylrhodamine acceptor (R) in the 3'-end) by the deoxyribozymes at its single ribonucleotide (rA). This substrate can anneal with the single stranded arms of the deoxyribozyme. When they anneal, the substrate splits into two oligonucleotides. One of them has a fluorescein donor (OF) and the other has a tetramethylrhodamine acceptor (OR). The reaction is initiated by the addition of Mg2+ (magnesium).[203]

Fig. 6.3(d) Schematic of the cleavage of E6 and 8-17
6.4 DNA based Logic Devices and their properties

Biomolecular computing is the computational method that uses the potential of DNA as a parallel computing device. DNA computing can be used to solve NP-complete problems. An appropriate application of DNA computation is large-scale evaluation of parallel computation models such as Boolean Circuits [202].

DNA followed the encoding principle of genetic information. This would require the formation and cleavage of numerous covalent bonds for their operation and thus require specific sets of enzymes.

Digital logic circuits create the base of computer architecture [207,208]. In constructing computer hardware different types of logic gates, e.g., OR, AND, XOR, NAND are utilized. In conventional Von Neumann machines, a program counter is used to perform in sequence the execution of instructions. These machines are based on a control-flow mechanism by which the order of program execution is explicitly stated in the user program. In a data flow machine the flow of the program is not determined sequentially, but rather each operation carried out as soon as the operands are ready [207].

One of the approaches suitable for massively parallel operations seems to be DNA-based molecular computing [209], with the help of DNA chips. These high density oligonucleotide DNA arrays were developed as tools for sequencing by hybridization [210]. Logical operations considered here are based on self-assembly of DNA strands. By self-assembly operation we understand putting together fragments of DNA during the process of hybridization [211].

Ogihara and Ray [211,212] suggested a DNA algorithm for implementing AND-OR basis Boolean Circuits that runs in time proportional to the size of the circuit. Their proposed algorithm works without exhaustive search. Amos et al. [213] described the first DNA based simulation of NAND Boolean circuits and improved the implementation to have run time that is proportional to the depth of the circuit. Ahrabian and Nowzari [214] proposed another algorithm for NAND circuits. They claimed that their algorithm is easier and the number of operation used is less than before. But they used error prone techniques such as PCR (Polymerase Chain
Reaction). Since then, all simulation models of Boolean Circuits has been constructed by OR and AND gates [201,211,216].

To build a computational system, first it is necessary to develop the simple parts of this system, as the logic gates. As discussed in the previous section of input and output strands, the schematic diagram can be shown as below Fig.6.4.

The catalytic oligonucleotides can be configured to exhibit a binary behaviour. When there is a cleavage of the substrate there is a “1” in the output because the oligonucleotide defined as output is one of the parts of the divided substrate. Then, when it is present in the final solution the output is interpreted as a “1”. When there is no cleavage, the output oligonucleotide is not produced and there is a “0” in the output.

The catalytic DNA only reacts when there is a concrete oligonucleotide present, if not it remains in inactive form. These two characteristics of deoxyribozyme are the reason why logic gates can be developed with it. The production of basic gates like YES (sensors), NOT, AND, OR, XOR, NAND and half-Adder have been discussed in this section.
6.4.1 DNA YES Gate

A YES or sensor gate transfers the detected input to the output[203]. Hence, the true table of this device is as follows:

<table>
<thead>
<tr>
<th>INPUT</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The deoxyribozyme E6 is chosen and a stem-loop is placed in the 5’-end. This stem-loop is configured with the complementary sequence of the input oligonucleotide. A solution containing this E6 configuration and the substrate, is prepared to receive the input.

If there is no input, the gate remains inactive because the closed loop overlaps with one of the arms of the catalytic oligonucleotide. Both arms together form the complementary sequence of the input. It means that if one of them is overlapped, the substrate can’t anneal with the E6. In this case, there is no cleavage and as a consequence there is no output oligonucleotide. There is a “0” in the output. If the desired input (the one complementary of the nucleotide sequence of the stem-loop) is added to the solution, the loop is opened and hybridizes with the input. Then, the
arms of the E6 are single stranded and anneal with its complementary which is the substrate. The cleavage is produced. The substrate is split into two oligonucleotides. The emissions of one of them that with the fluorescenin donor are detected in the fluorescence spectra. There is a “1” in the output.

6.4.2 DNA based OR Gate

An OR is a gate with two inputs and one output. It sets the output to “1” when there is a binary “1” in at least one of its inputs. Its truth table is as follows:

<table>
<thead>
<tr>
<th>INPUT A</th>
<th>INPUT B</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

![Fig. 6.4.2 DNA based OR gate](image)

An OR gate is constructed with two kinds of (YES) sensor gates. The E6 deoxyribozyme is used to build an OR gate. The input of each of these groups has a different sequence of nucleotides. The two kinds of sensor gates are put together with a substrate in a solution.

When at least one group of the gates is activated by its respective input, the cleavage of the substrate happens. Therefore, the oligonucleotide output with the fluorescenin donor is present in the solution. This means a “1” in the output. In this case, the desired loop is opened and hybridizes with the original input. Then, the [395]
arms of the E6 are single stranded and anneal with its complementary which is the substrate. The cleavage is produced. The substrate is split into two oligonucleotides. The emissions of one of them that with the fluorescein donor are detected in the fluorescence spectra. There is a “1” in the output.

When both inputs are missing, the cleavage doesn’t occur and the expected output oligonucleotide is not present. The output is “0”. If there is no input, the gate remains inactive because the closed loop overlaps with one of the arms of the catalytic oligonucleotide. Both arms together form the complementary sequence of the input. It means that if one of them is overlapped, the substrate can’t anneal with the E6. In this case, there is no cleavage and as a consequence there is no output oligonucleotide. There is a “0” in the output.

### 6.4.3 DNA based NOT Gate

A NOT gate puts in the output the opposite bit to the input. The true table is as follows:

<table>
<thead>
<tr>
<th>INPUT</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The implementation with DNA is made with E6 but changing the structure with regard to the one used as YES gate. The nonreacting loop of E6 is substituted by a stem-loop with a sequence complementary to the input. The gate and the substrate are put together in a solution. If the input is added to the solution (a one in the input),
the hybridization between the input and the stem-loop with complementary sequence produces the changing of the deoxyribozyme into its inactive form. As a result, there is no cleavage and therefore, neither is output oligonucleotide. This means a “0” in the output. But if no input is added, the E6 remains in its active form and cleaves the substrate, producing the output oligonucleotide (output is “1”).

6.4.4 DNA based AND Gate

An AND gate is a two input gate that only sets the output to “1” when both inputs are “1”. The true table of this device is:

<table>
<thead>
<tr>
<th>INPUT A</th>
<th>INPUT B</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The 8-17 deoxyribozyme is used to build an AND gate. Two stem-loops, one in the end of each arm of the oligonucleotide, are joined to the 8-17 structure. The sequence of nucleotides of these loops is different. Thus, each one reacts with a different input (the one that has its complementary sequence). If both inputs are added to the solution with the gate and the substrate, they hybridize with the stem-loops. The stem-loops are opened and the substrate anneal with its complementary oligonucleotide in the deoxyribozyme. The cleavage of the substrate produces the output oligonucleotide. The output is set to “1”.

[397]
If only one of the inputs is added, its complementary stem-loop is opened but the other doesn’t. The one that remains closed overlaps one of the parts of the catalytic oligonucleotide that must anneal with the substrate to cleave it. This way, the cleavage does not occur and there is no output oligonucleotide: “0” in the output. If none of the inputs are put in the solution, both stem-loops of the gate are closed. They overlap the part of the deoxyribozyme that has to anneal with the substrate to cleave it. No cleavage occurs and therefore the output oligonucleotide does not exist. There is a “0” in the output.

6.4.5 DNA based XOR Gate

A XOR gate is a two input gate. It sets to “1” the output when one and only one of the inputs is “1”. Then, the true table is as follows:

<table>
<thead>
<tr>
<th>INPUT A</th>
<th>INPUT B</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig.6.4.5 (a) one of the group of DNA based XOR gate

The XOR gate is the most difficult to implement because the gate must be active when one of the inputs is present, it does not matter which of them. But when both of them are present, the gate must be inactive. Then, sometimes must be active with a certain input and other times must be inactive with the same input when it is attached to the other input.
The XOR gate is composed by two groups of gates. There are two input oligonucleotides. In order to make a clear explanation, one will be referred to as “the first” and the other as “the second”. One of the groups is active when only the first input is in the solution but not the second. The other group must be activated by the presence of only the second group but not the first. Therefore, when both inputs are in the solution none of the groups of gates is active. Each gate of one of the two groups is a mix between a YES (sensor) gate and a NOT gate.

The group which is active with only the first input is formed by E6 deoxyribozymes ended by a stem-loop. This stem-loop has a sequence of nucleotides complementary to the sequence of the first input. It behaves like a sensor for the first input. These gates have another stem loop in the core of the deoxyribozyme. It works with the second input like the loop in the NOT gate.

The translation to logic language will be $A \wedge \neg B$ (A is the first input and B is the second).
CHAPTER-6
DNA Based Computing Devices

This happens analogously for the other group of gates that in logic language will be $B^\land \neg A$. Then, if only one input is present in the solution, one group of gates is inhibited and the other is in active form. Thus, there is cleavage of the substrate and output oligonucleotide in the final solution. But if both inputs are present the two groups are inactive. No cleavage and hence no presence of the output oligonucleotide. Similar events happen when there are no inputs. This behavior can be translated like an OR (Fig. 6.4.2) of the functions carried out by the two groups $(A^\land \neg B \lor B^\land \neg A)$.

6.4.6 DNA based NAND Gate

A NAND is a two inputs gate that sets the output to "0", when both inputs are "1". In the rest of the cases the output is "0". Then, the true table is as follows:

<table>
<thead>
<tr>
<th>INPUT A</th>
<th>INPUT B</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig.6.4.6 DNA based NAND Gate

A NAND gate is a solution containing two groups of NOT gates with different structure. There is a different sequence of nucleotides in the input of one group from the one of the other.

When both inputs are added to the solution both gates are inhibited and no output is produced ("0"). If one of the inputs is missed in the solution, one of the gates produces output oligonucleotide by splitting the substrate and if both are missed the
two gates produce output oligonucleotide by splitting the substrate. In this case, the output is “1”.

6.4.7 DNA based NOR Gate

A NOR is a two inputs gate that sets the output to “1”, when both inputs are “0”. In the rest of the cases the output is “0”. Then, the true table is as follows

<table>
<thead>
<tr>
<th>INPUT A</th>
<th>INPUT B</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig.6.4.7 DNA based NOR Gate

A NOR gate is a solution containing two groups of NOT gates with different structure. There is a different sequence of nucleotides in the input of one group from the one of the other. It is produced by taking into consider 8-17 structure.

1- 1-condition: When both inputs are added to the solution both gates are inhibited and no output is produced (“0”). This way, the cleavage does not occur and there is no output oligonucleotide: “0” in the output. They overlap the part of the deoxyribozyme that has to anneal with the substrate to cleave it. No cleavage occurs and therefore the output oligonucleotide does not exist. There is a “0” in the output.

1-0, 0-1 condition: If one of the inputs is missed in the solution, one of the gates produces output oligonucleotide by inhibiting the substrate so the output is “0”. In this case if only one of the inputs is added, its complementary stem-loop is opened but the other doesn’t. The one that remains closed overlaps one of the parts of the catalytic [401]
oligonucleotide that must anneal with the substrate to cleave it. This way, the cleavage does not occur and there is no output oligonucleotide: “0” in the output.

**0-0 condition:** If both are missed the two gates produce output oligonucleotide by splitting the substrate. In this case, the output is “1” i.e if both inputs are added to the solution with the gate and the substrate, they hybridize with the stem-loops. The stem-loops are opened and the substrate anneal with its complementary oligonucleotide in the deoxyribozyme. The cleavage of the substrate produces the output oligonucleotide and output is set to “1”.

### 6.4.8 DNA based Half-Adder

A half adder is a more complex circuit generally considered as combinational circuit where it uses fundamental logic gate binary addition principle. It gives an output consisting of two bits. One of them is the binary addition of the two inputs as a result of a XOR operation between the two bits and the other bit is the carry which can be implemented as the AND operation over the two inputs [McC86]. This signifies as SUM and CARRY operation.

1) T-C·A-C·T-A·T-rA·G·G·A·A·G·A·G
2) T-C·A·A·G·C·C·rA·A·A·C·C·G·A·G

**Fig.6.4.8 (a) Substrate of the Half-Adder**

The AND and XOR gates are designed to accept the same input oligonucleotides, that has discussed earlier **Fig.6.2 (c)**. But they act over different substrates.

The substrate (**Fig.6.4.8 (a)**) can be cleaved by the XOR gate **Fig.6.4.8 (b)** that makes the sum module 2 of the inputs. The substrate can be cleaved by the AND gate **Fig.6.4.8 (C)** that obtains the carry.
Fig. 6.4.8 (b) SUM from XOR

Both gates (XOR and AND) are mixed with the substrates in a solution. To sum 1+1, both input oligonucleotides are added to the solution. The XOR gates become in inactive form and AND gates become in active form (they cleave the pertinent substrate, as has been explained). In the final solution only $O_F$ (Fig. 6.4.8(c)) is present. The binary result is “10” (2 in decimal). To sum 1+0 or 0+1, only one input oligonucleotide is added and then only the XOR is active and cleaves its substrate. This way, the $O_F$ (Fig. Fig. 6.4.5 (a)) is present in the final solution. The result in binary is “01” (1 in decimal). To sum 0+0, no input nucleotides are added. No gate is active; there is no cleavage and no output oligonucleotides. Therefore, the result is “00”. 
6.5 DONOR and ACCEPTOR Properties Analysis

As it has discussed in above sections, after the logic operation, there are two possible outputs occurred. When the cleavage happens, there is output oligonucleotide. This is interpreted as “1”. But if there is no cleavage, no output oligonucleotide is generated. There is a “0” in the output.

The substrate (Fig.6.5(a)) is an oligonucleotide ended with a fluorescein donor (F) at the 5’-end and with a tetramethylrhodamine acceptor at the 3’-end. This configuration was chosen because when this substrate is cleaved at the single ribonucleotide (rA) the fluorescence of the substance is incremented. If cleavage has not happened, when a beam of electrons of \( \lambda = 480 \text{nm} \) is sented through fluorescence spectrum method [Vam93], the acceptor (R) absorbs much part of the emission of the donor. As it is possible to see in Fig.6.5 (b) (brown line), there is not a peak of emission in. The output is “0”. But if cleavage has taken place Fig.6.5(c) when the beam of electrons of \( \lambda = 520 \text{nm} \) is sended, the donor emits at \( \lambda = 480 \text{nm} \). On scanning from \( \lambda = 500 \text{nm} \) to \( \lambda = 600 \text{nm} \), a peak at \( \lambda = 520 \text{nm} \) is detected (Fig.6.5 (b), blue line). The output is “1”. This method is only valid to detect one output (one bit), but not valid for more output bits (e.g. half adder).
The recent discovery by Isenberg and Szent-Györgyi [216] for the possible important biological implications of electron-donor and electron-acceptor properties of biologically important substances like purines, pyrimidines, etc has been added a new approach to find the electron transport properties and conductivity in a biological molecule. As no experimental data seem to be available about the molecular
ionization potentials or electro affinities of these molecules, it has taken quantum-mechanical calculations of the distribution of electronic energy levels to obtain the corresponding information [217].

Using molecular-orbital method the energies of the molecular orbitals of the mobile or \( \pi \) electrons of the system is given by \( E_i = \alpha + K_i \beta \), where \( \alpha \) is the coulomb integral and \( \beta \) is the resonance integral of the method. Positive values of \( K_i \) correspond to occupied (bonding) orbitals, negative values of \( K_i \) to empty (antibonding) orbitals. The smallest positive value of \( K_i \) corresponds to the highest occupied molecular orbital (HOMO.)[217], and the comparison of the value of this parameter in a series of related compounds gives the relative value of their ionization potentials: the smaller the value, the lower the ionization potential and consequently the greater the electron donor capacity of the molecule. The smallest negative value of \( K_i \) corresponds to the lowest empty molecular orbital or lowest unoccupied molecular orbital (LUMO.), and the comparison of the value of this parameter in a series of related compounds measures the relative value of their electro affinity: the smaller the value, the greater the electro affinity [217].

The below table shows the HOMO and LUMO level energy of DNA nitrogen bases.

<table>
<thead>
<tr>
<th>Compound</th>
<th>Energy of Highest Occupied Molecular Orbital</th>
<th>Energy of Lowest Empty Molecular Orbital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adenine</td>
<td>0.468</td>
<td>-0.865</td>
</tr>
<tr>
<td>Guanine</td>
<td>0.307</td>
<td>-1.050</td>
</tr>
<tr>
<td>Cytosine</td>
<td>0.595</td>
<td>-0.795</td>
</tr>
<tr>
<td>Thymine</td>
<td>0.510</td>
<td>-0.958</td>
</tr>
<tr>
<td>Uracil</td>
<td>0.597</td>
<td>-0.960</td>
</tr>
</tbody>
</table>

Table:6.5 Orbital energy for Nitrogen bases of DNA Molecule

The best electron donor among these bases is guanine. A simple way of increasing the electron-donor properties of purines is N-methylation. The degree to which such methylation increases these properties depends on the compound and, for a given compound, on the position of the alkyl group. Pyrimidines, i.e., uracil, thymine, and cytosine are only moderate donors and moderate acceptors. In particular, Pyrimidines are poorer donors than the fundamental purines. The coupling of purines
and pyrimidines through hydrogen bonding such as the one which occurs in the nucleic acids, following the Watson and Crick model. This implies the electron-donor properties of the purines: the energy of the HOMO of the adenine (A)-thymine (T) couple is at 0.425, while that of the guanine (G)-cytosine(C) couple is at 0.308. [217]

6.6 Advantage and Disadvantage of DNA based Computing System using Deoxyribozyme

The advantage of DNA model computing over silicon can be said as the parallelism approach in executing processes [213], the low power consumption and the high speed of the molecular events. The disadvantage in terms of human intervention is necessary in the computations, access to the results and the data is slow.

As discussed above, Deoxyribozyme lets implement all the logic gates and therefore it is possible to perform all the functions that can be expressed like a combination of these basic logic functions. The inputs and outputs are oligonucleotides of same nature. Hence, the outputs of a gate can be used as input of other gates to construct a system. This feature is not so realizable with silicon technology, but can realizable for molecular computing.

The deoxyribozyme as a catalyst that can model with any nucleic acid sequences of sufficient length as input and can increase the degrees of freedom. New discoveries and researches are made, and the knowledge about the possibilities of the deoxyribozyme, as material to perform computations is not so prominently well-known like the enzymes. Therefore, thus future technology seems to be encouraging and challenging to give computational power to system using biomolecules.

6.7 Bimolecular Computation

Bimolecular computation (BMC) has the potential to store huge memories. Each individual strand of DNA can encode binary information. A small volume can contain huge number of molecules. BMC has also high the potential to supply massive computational power. In the construction of a parallel machine, each processor’s state is encoded by a DNA strand. BMC can perform massive parallel computation by recombinant DNA operations that act on all DNA molecules at the same time. 5 gms of DNA in a liter of water contain $10^{21}$ bases. For an associate memory the some
bytes of memory can be formed with 100 base pairs per DNA strand. Thus a liter of solution provides $10^{19}$ to $10^{20}$ bytes which is $10^7$ to $10^8$ terabyte. A DNA strand may need 1000 base pairs to encode a processor state. A liter of solution encodes the state approximately $10^{18}$ distinct processors. The time duration of recombinant operation such as annealing which occurs through hybridization is depend on the length of the sequence to be matched. This biological process takes 100 minute. The overall operation using BMC takes 1015 to 1020 operations per second. It talks 10-19 joules of energy per operation [218].

The DNA memory generally represents in write and read form. The write form is nothing but how does one form the sequence and reading is the hybridization result.

![Figure 6.7 Memory in DNA Molecule](image)

[408]
6.8 Difference between Silicon Based Computer and DNA-based Computer

The difference between DNA-based computer and silicon based computer depend on the device design and operation.

<table>
<thead>
<tr>
<th>DNA-based computers</th>
<th>Microchip-based computers</th>
</tr>
</thead>
<tbody>
<tr>
<td>slow at individual operations</td>
<td>fast at individual operations</td>
</tr>
<tr>
<td>can provide huge memory in small space</td>
<td>smaller memory</td>
</tr>
<tr>
<td>can do billions of operations simultaneously</td>
<td>can do substantially fewer operations</td>
</tr>
<tr>
<td>setting up a problem may involve considerable preparations</td>
<td>setting up only requires keyboard input</td>
</tr>
<tr>
<td>DNA is sensitive to chemical deterioration</td>
<td>electronic data are vulnerable but can be backed up easily</td>
</tr>
</tbody>
</table>

6.9 Current trends of DNA Computing

After Adleman’s experiment, the interest in DNA computing increase worldwide. Many theories and experiments have been done by scientists about the computational capabilities of DNA [219]. It opened a discussion about the real applications of molecular computing. The molecular technology is now all total a separate specific field looking beyond silicon technology.

The evolution of the DNA computing makes us realize the well known biological processes and events that responsible for computational power. A normal DNA does not react alone, it reacts in association with a catalyst called enzyme. The new model of computing based on catalytic oligonucleotides has established new ways to deal with DNA computing and has validated the theoretical possibilities with an experimental implementation [220]. A complete set of logic gates is developed by means of molecular devices with DNA as the ingredient. The future of the DNA based model is the possible in years to come. Only time can point out the real possibilities of this sequences of nucleotides with such special characteristics.
Summary

This chapter has been discussed about the fundamental DNA computing process and various molecular logic devices designed from DNA. This chapter has taken a study of DNA based logic gates like YES, NOT, AND, OR, XOR, NAND, NOR and Half-Adder gates. The specification given in this chapter is only to compare the designing and operation principle of DNA based computing. The comparative study of designing and computing principle of silicon based computing system, organic based computing system and DNA based computing system has been discussed in the next chapter.