Implementation of Data Flow Logical Operations via Self-Assembly of DNA

Piotr Wąsiewicz[1)], Piotr Borsuk[2)], Jan J. Mulawka[1)], Piotr Węgleński[2)]

[1)]Institute of Electronics Systems, Warsaw University of Technology
Nowowiejska 15/19, Warsaw, Poland
{jml,pwas}@ipe.pw.edu.pl
[2)]University of Warsaw, Pawińskiego 5A, Warsaw, Poland
wegle@ibb.waw.pl

Abstract: Self-assembly of DNA is considered a fundamental operation in realization of molecular logic circuits. We propose a new approach to implementation of data flow logical operations based on manipulating DNA strands. In our method the logic gates, input, and output signals are represented by DNA molecules. Each logical operation is carried out as soon as the operands are ready. This technique employs standard operations of genetic engineering including radioactive labeling. To check practical utility of the method a series of genetic engineering experiments have been performed. The obtained results confirm interesting properties of the DNA-based molecular data flow logic gates. This technique may be utilized in massively parallel computers.

1 Introduction

Digital logic circuits create the base of computer architecture. Digital systems consist sometimes of single logic gates and switches [1]. In constructing computer hardware different types of logic gates e.g. OR, AND, XOR, NAND are utilized. Since computers are very important in our life and more powerful machines are still needed it is worth considering various implementations of their architectures. 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.

Data flow machines do not fall into any of the classifications described above. In a data flow machine the flow of the program is not determined sequentially, but rather each operation is carried out as soon as the operands are ready. This data-driven mechanism allows the execution of any instruction depending on data availability. This permits of a high degree of parallelism at the instruction level and can be used in recent superscalar microprocessor architectures. However, in the field of parallel computing, dataflow techniques are in the research stage and industry has not yet adopted these techniques[3]. One of approaches suitable for massively parallel operations seems to be DNA based molecular computing [4].

The primary objective of this contribution is to show striking adequacy of molecular computing for dataflow techniques. 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 [5]. Since processing units as well as input and output signals are represented by hybridizing DNA strands, data-driven mechanisms of performance are assured. We present a new approach to implementation of logical operations suitable for dataflow architectures. Our method is an extension of works reported by Ogihara, Ray and Amos [6, 7, 8].

2 Computing Properties of DNA Molecules

A number of papers have appeared on general concept of molecular computers [9, 10, 11, 12]. In such approach computations are performed on molecular level and different chemical compounds can be utilized. Deoxyribonucleic acid (DNA) is usually used in molecular computing because this compound serves as a carrier of information in living matter and is easy to manipulate in genetic engineering laboratory.

The DNA molecule is composed of definite sequence of nucleotides. There are four different nucleotides in such molecule depending on purine and pyrimidine bases present in the given nucleotides, which we denote by the letter: A, T, C, G (adenine, thymine, cytosine, guanine). Thus the DNA molecule can be considered as a polynucleotide or a strand or a string of letters. Since the strands may be very long, they can carry an immense number of bits. Therefore particular molecules may serve as information carriers.

Another interesting property of DNA is that particular bases join together forming pairs: A with T and C with G. This process known as hybridization or annealing causes self-assembly of DNA fragments. It occurs when two DNA strands are composed of matching (complementary) nucleotide sequences. Due to favored intermolecular interactions particular molecules can recognize each other. As a result a kind of key-lock decoding of information is possible. The self-assembly property may be utilized to implement the memory of an associate type. According to Baum [13] this memory may be of greater capacity than that of human brain. The other important feature of self-assembling is that particular molecules may be considered as processing units performing some computing.

The following techniques of the genetic engineering will be utilized in our approach: synthesis of DNA strands, labeling, hybridization, analysis by DNA electrophoresis, digestion of double strands by restriction enzymes, synthesis of DNA strands by Polymerase Chain Reaction (PCR) [14, 15].

3 Dataflow Parallel Processing Units

In a dataflow machines [16], data availability rather than a program counter is used to drive the execution of instructions. It means that every instruction that has all of its operands available can be executed in parallel. Such an evaluation of instructions

permits the instructions to be unordered in a data-driven program [3].

Instead of being stored in a shared memory, data are directly hold inside instructions. Computational results are passed directly between instructions via data tokens. A single result is forwarded via separate tokens for each instruction that needs it. Results are not stored elsewhere and are not available to instructions that do not receive tokens. The asynchronous nature of parallel execution with the data-driven scheme requires no shared memory and no program counter. However, it implies the need for special handshaking or token-matching operations. If a pure dataflow machine could be cost-effectively implemented with a low instruction execution overhead, massive fine-grain parallelism at the instruction level could be exploited.

The computation is triggered by the demand for an operation's result. Consider the evaluation of a nested arithmetic expression z=(x+y)*(w-5). The data-driven computation chooses a bottom-up approach, starting from the innermost operations x+y and w-5, then proceeding to the outermost operation *. Such operations are carried out immediately after all their operands become available.

In a data flow processor the stored program is represented as a directed graph as shown for example in Fig. 1. The graph is reduced by evaluation of branches or subgraphs. Different parts of a graph or subgraph can be reduced or evaluated in parallel upon demand. Nodes labeled with particular operations calculate a result whenever the inputs are valid, and pass that result on to other nodes as soon as it is valid which is known as firing a node. To illustrate this operation some process of computation is described in Fig. 2. As is seen a node representing a performed process has two incoming branches, which represent inputs. The flow of data is shown in these pictures by introducing data tokens denoted by block dots. In Fig. 2a there is a moment when data on a single input has appeared and the processor is waiting for a second data token while in Fig. 2b the processor is ready to execute its function and finally in Fig. 2c there is a situation after firing the node. The dataflow graphs usually are more complicated and there are different types of operations applied. Note that there are no variables in a data flow program, and there can be no notion of entities such as a program counter or global memory.

Generally, molecular logic gates may be utilized in constructing computer architecture based on dataflow graphs. Such graphs consisting of DNA strands can exchange tokens as DNA single strings representing gate inputs and outputs. Some sectors of strands can transport values of variables or special constants. In such solution a number of gates-nodes (processing units) and a number of data and control tokens may be theoretically illimitable. In the next point we demonstrate how to realize data flow logical processing units using DNA molecules.

4 A New Concept of Data Flow Logic Gates

In classical computer the logic gate is an electronic circuit that has one or more inputs and one output. In such circuit the electrical condition of the output at any time is dependent on those of the inputs at that time [1]. An alternative approach to implement logic operations may be performed on molecular level by self-assembly of DNA

strands. In such method DNA molecules as is depicted in Figs 3a respectively represent the logic gates, input and output signals.

Assume that strands representing gates consist of three sectors belonging to the following groups: $\underline{X}=\{\underline{x}_1, \underline{x}_2, \underline{x}_3,...,\underline{x}_N\}$, $\underline{Z}=\{\underline{z}_1, \underline{z}_2, \underline{z}_3,...,\underline{z}_N\}$, $\underline{Y}=\{\underline{y}_1, \underline{y}_2, \underline{y}_3,...,\underline{y}_N\}$. Representing inputs strands are composed of two sectors from groups: $X=\{x_1, x_2, x_3,...,x_N\}$, $Y=\{y_1, y_2, y_3,...,y_N\}$. Representing outputs strands consist of three sectors belonging to groups: $X=\{x_1, x_2, x_3,...,x_N\}$, $Z=\{z_1, z_2, z_3,...,z_N\}$, $Y=\{y_1, y_2, y_3,...,y_N\}$. The composition of DNA strands are depicted in Fig. 3b. To enable hybridization process, input and output signals sectors from the groups X, Z, Y are complementary to adequate sectors in logic gates strands: $\underline{X}$, $\underline{Z}$, $\underline{Y}$. This means that two DNA strands can be connected together with the two sequences in the following way: x_i with $\underline{x}_i$ or z_i with $\underline{z}_i$ or y_i with $\underline{y}_i$. There must be one sequence: x_i or z_i or y_i in the first string and another complementary sequence: $\underline{x}_i$ or $\underline{z}_i$ or $\underline{y}_i$ in the second string. The mentioned two sequences hybridize each other forming a double-strand fragment.

Thus the sectors x_1 and x_2 of input I_1 and I_2 strands pair with matching sections ($\underline{x}_1$ and $\underline{x}_2$) in the AND strand. Output signals are composed of input strands and other DNA fragments. These fragments ($\underline{z}_i$) are reserved to accomplish special tasks. For example they can transport additional information about logical gates, nodes in a computer program etc. The complete molecules after annealing are shown in Fig. 3c and Fig. 4c. Upper strand is labeled. Labeling of this strand can be achieved by introducing radioactive, biotinylated or fluorescent nucleotide. In our experiments (see section 6) we have employed the technique of filling the gap (A^*) left between the sectors (oligonucleotides) z_1 and x_2 by addition of ^{32}PdATP using the Klenov polymerase. All strands are ligated with DNA ligase and then the output strands are disconnected from gate strands as is shown in Fig. 3d and then analyzed using standard method of the DNA electrophoresis in acrylamide gel. Single stranded radioactive oligonucleotides representing output of the logic gate are detected by autoradiography. Fragments of proper length could be isolated from mentioned gel and used as signal and data tokens in the next layer of molecular gates. As follows from Fig. 3 both AND and OR gates have similar construction, but when OR gates have two or more inputs, then two or more different molecules each with one input are created.

5 DNA-Based Implementation of Simple Logic Network

A Boolean function is composed of binary variables, operation symbols, brackets and a symbol of equivalence relation. The Boolean function describes usually more complicated logic networks composed of logic gates. Some number of connected logic gates with sets of inputs and outputs is called a combinational network. However, in a sequential network there are not only logic gates, but also flip-flops with memory [1].

We have considered to implement simple combinational network by operations on DNA strands. An example of a simple combinational network and its implementation in DNA as well as its performance is shown in Fig. 4. The logic gates are DNA oligonucleotides shown in Fig. 4b. Two gates AND_1, AND_2 are self-assembling molecules from Fig. 4c, d. In Fig. 4e these two molecules anneal to DNA strand of the

AND_3 gate. The regions named A^* are for adding labeled radioactive adenine by the Klenov polymerase to a 3' end of oligonucleotides as described in Fig. 3. It should be noted that $E^{1,2}$ restriction sites are ready for digestion if there is a double DNA strand there. After hybridization and ligation a process of digestion a double strand molecule by restriction enzymes is performed. The enzymes cut the double strand in areas E^1 and E^2 yielding three fragments from which output strands could be potentially rescued after strand separation. It is necessary to remove enzymes and redundant strings before further computation. The redundant strings are removed in a process of electrophoresis.

Neighbouring logic gates layers could be connected with each other to form the combinational network. A number of layers may be theoretically illimitable if the oligonucleotides are recycled during computing after an execution of some number of layers. For construction of our molecular gate layers one can envisage use of solid support. These gates can be for example attached to DNA chips [17]. E.g. if in Fig. 4f the AND_3 gate strand was attached to the DNA chip and the AND_1 and AND_2 gate strands – to the magnetic beads with biotin, then output strings could be easily separated.

6 Results of Experiments

To confirm oligonucleotide self-assembling as a technique useful in data flow logical operations several experiments were performed.

Four oligonucleotides were designed for self-assembling. The lower strand is composed of one longest oligonucleotide named AND (30 bp) – 5' AGTAACTCCCTCACCATCGCTCGCCAACTC 3' and the upper strand of three other: I1 – 5' GGTGTCGATCATTGAG 3', SR – 5' GGAGTGGTAGCG 3' and I2 – 5'CGGTTGAGCTTGGAAG 3'. In our experiments I1 and/or I2 were hybridized to AND in presence of SR oligonucleotide. Typically 5 – 10 pM of each oligonucleotide were used and hybridization was performed in ligase buffer for 5 – 30 min. in 40^0C. Under these conditions formation of the SR/AND hybrid was preferred. In next step 5 μCi of ^{32}PdATP (3000 Ci/mM, Amersham-USB), 100 pM dGTP and 10u of Klenov polymerase (Amersham-USB) were added, and samples were incubated for 30 min at 37^oC. Afterwards the temperature was lowered to 25^oC (to hybridize I1 and/or I2 to AND) and samples were incubated for 30 min. 5u of ligase (Promega) were added. Ligation were performed at 25^oC for 2 h, and at 14^oC for 2 – 24 h. Ligation reactions were terminated by addition of the equal volume of the stop mixture (formamide with 0,05% bromophenol blue and 0,05% xylene cyanol). Samples were denatured by 5 min. incubation in 80^0C and loaded onto poliacrylamide (8 or 20%) denaturing (8M urea) gel. Electrophoresis was performed under standard conditions. Radioactive single stranded DNA were detected in electrophoretograms by autoradiography (Fig. 5 and 6). Typically only proper single stranded DNA fragments were obtained (Fig. 5). Oligonucleotide self-assembling was not disturbed by addition of the random oligonucleotide (100 pM per reaction) (Fig. 6 A1, B1 and C1).

7 Conclusions

We have explored the possibility of implementation of data flow logical operations by DNA manipulations. We have demonstrated that self-assembly of DNA can be utilized to provide the flow of data. In our approach standard genetic operations are employed. An appropriate DNA reactions have been performed and their results confirmed our assumptions. We think that our methodology has several advantages when compared with that proposed by Ogihara and Amos [6,7,8] and can be further improved by the use of DNA chips. This would permit for easy separation of output and input strands and would provide an important step towards massively parallel molecular computer.

References

[1] Biswas, N.N.: Logic Design Theory, Prentice-Hall International Editions, USA, 1993

[2] Hill, F. J., Peterson, G.P.: Switching Theory and Logical Design. Wiley New York (1974)

[3] Landry E., Kishida Y.: A Survey of Dataflow Architectures. Internet.

[4] Mulawka, J. J., Wąsiewicz, P.: Molecular Computing, (in Polish) Informatyka, 4, April (1998) 36-39

[5] Mulawka, J.J., Borsuk, P., Węgleński, P.: Implementation of the Inference Engine Based on Molecular Computing Technique. Proc. IEEE Int. Conf. on Evolutionary Computation (ICEC'98), Anchorage USA (1998) 493-496

[6] Ogihara M. and Ray, A.: Simulating Boolean Circuits On a DNA Computer. Technical Report TR 631, University of Rochester, Computer Science Department, August (1996)

[7] Ogihara M. and Ray, A.: The Minimum DNA Computation Model and Its Computational Power. Technical Report TR 672, University of Rochester, Singapur (1998)

[8] Martyn Amos and Paul E. Dunne. DNA Simulation of Boolean Circuits. Technical Report CTAG-97009, Department of Computer Science, University of Liverpool UK (1997)

[9] Adleman, L.M.: Molecular Computation of Solutions to Combinatorial Problems. Science, vol. 266. (1994) 1021-1024

[10] Adleman, L.M.: On Constructing a Molecular Computer, Internet

[11] Kurtz, S., Mahaney, S., Royer, J.S., Simon, J.: Biological Computer, Internet

[12] Dassen, R.: A Bibliography of Molecular Computation and Splicing Systems, at the site WWW http://liinwww.ira.uka.de/bibliography/Misc/dna.html

[13] E.B. Baum, Building an Associative Memory Vastly Larger than the Brain, Science, vol. 268, 583-585, 1995.

[14] Sambrook, J., Fritsch, E.F., Maniatis, T.: Molecular Cloning. A Laboratory Manual. Second Edition, Cold Spring Harbor Laboratory Press (1989)

[15] Amos, M.: DNA Computation. PhD thesis, Department of Computer Science, University of Warwick UK, September (1997)

[16] Papadopoulos G.M.: Implementation of a General Purpose Dataflow Mulitprocessor. PhD thesis, MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139, August 1988.

[17] Wodicka, L., et al.: Genome-wide Expression Monitoring in Saccharomyces cerevisiae. Nature Biotechnology 15 Dec. (1997) 1359-1367

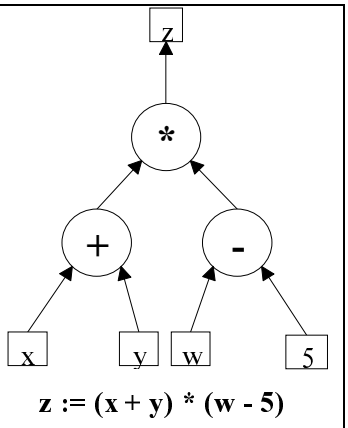

Fig. 1. A dataflow graph of a simple program

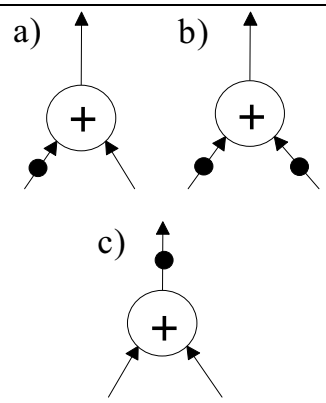

Fig. 2. Successive phases of firing a node: a) waiting for a second data token, b) ready to execute an operation, c) after firing

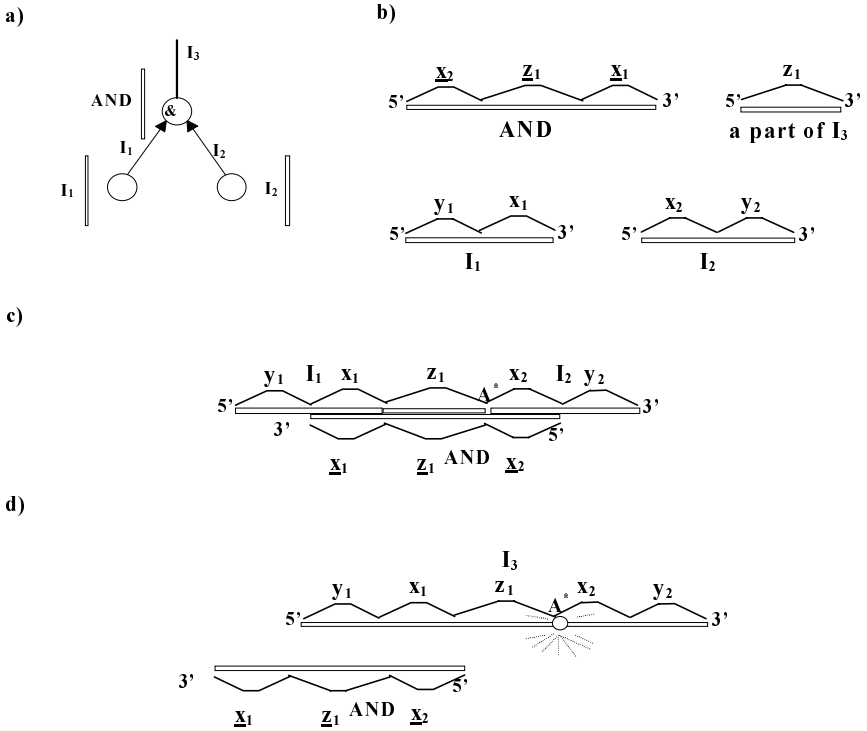

Fig. 3. A data flow logic AND gate: a) a scheme of the gate, b) oligonucleotides representing the AND gate, input signals I_1, I_2 and a part of an output I_3, c) hybridized oligos, d) disconnecting of the AND gate and an output I_3; A^* - regions of adding labeled radioactive Adenine by Klenov polymerase to a 3' end of z_1, so there must be several free Tymine nucleotides in the AND gate strand in the area between $\underline{z}_1$, I_2.

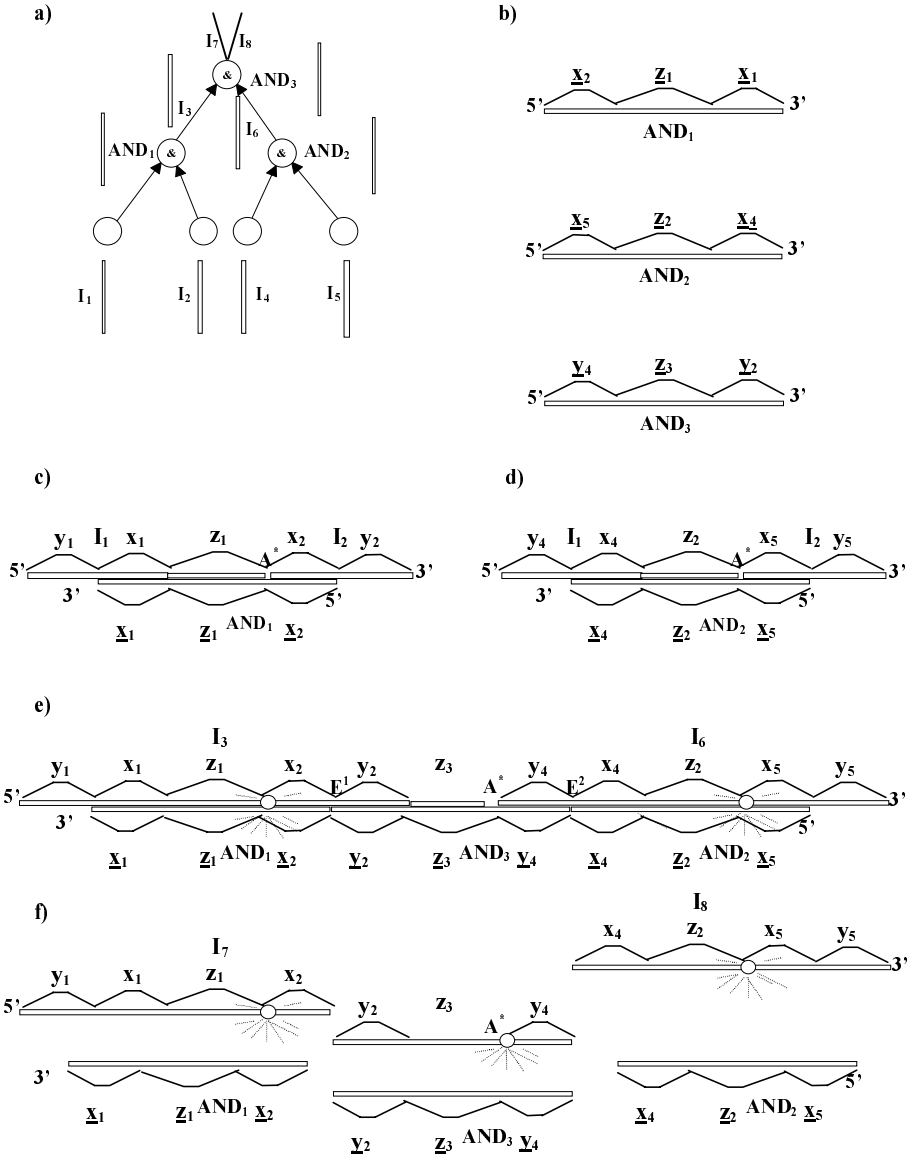

Fig. 4. A data flow combinational network: a) a scheme of the logic system, b) oligonucleotides representing logic gates, c) and d) hybridized oligonucleotides of gates AND_1 , AND_2 with oligos of inputs signals and parts of output signals, e) hybridized an oligonucleotide of the AND_3 gate with hybridized earlier oligos of gates AND_1 , AND_2, f) digestion by restriction enzyme of the restriction sites E^1, E^2, disconnecting of the AND gates and outputs I_7 and I_8; A^* regions of adding labeled radioactive Adenine by Klenov polymerase to a 3' end of oligonucleotides as described in Fig.1 and Fig.2. ;$E^{1,2}$ restriction sites ready for digestion if there is a double DNA strand there.

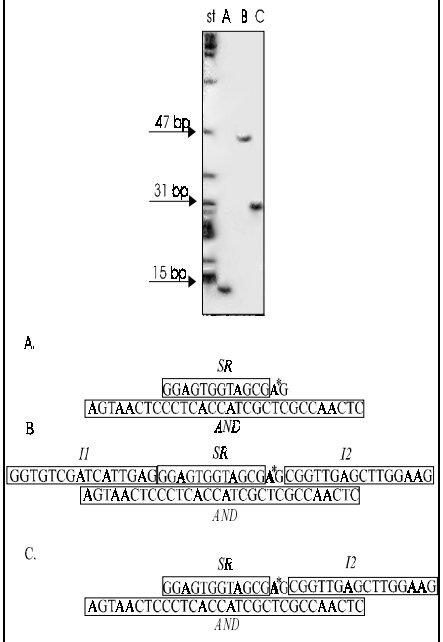

Fig. 5. Autoradiogram of ligation products. Ligations products (single stranded DNA were analysed by electrophoresis in denaturing 8% polyacrylamide gel. st - molecular weight standard, A - oligonucleotide SR was hybridized to oligonucleotide AND and radioactive nucleotide (A*) was added to 3' end of oligonucleotide SR by Klenov polimerase, B - self-assembly of oligonucleotides I1, SR, I2 by hybridization to oligonucleotide AND, radioactive A (A*) were added by Klenov polymerase, C - self-assembly of oligonucleotide SR and I2 by hybridization with oligonucleotide AND, radioactive A (A*) was added by Klenov polymerase.

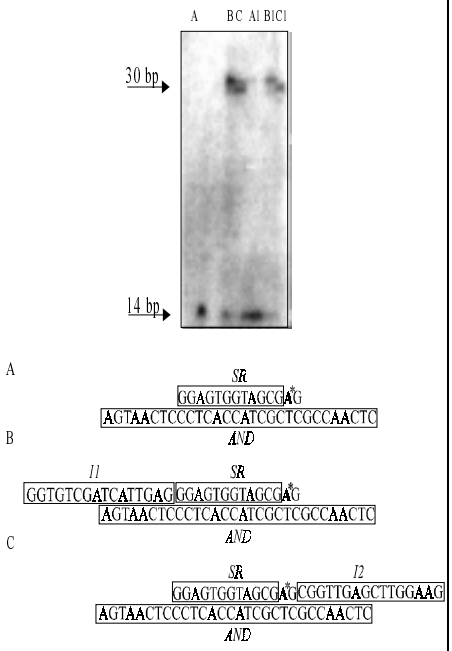

Fig. 6. Autoradiogram of ligation products. Ligations products (single stranded DNA were analysed by electrophoresis in denaturing 20% polyacrylamide gel. A - oligonucleotide SR were hybridized to oligonucleotide AND and radioactive nucleotide (A*) was added to 3' end of oligonucleotide SR by Klenov polimerase, B - self-assembly of oligonucleotides I1 and SR by hybridization with oligonucleotide AND, radioactive A (A*) was added by Klenov polymerase, C - self-assembly of oligonucleotide SR and I2 by hybridization with oligonucleotide AND, radioactive A (A*) was added by Klenov polymerase. A1, B1, C1 - self-assembly and labeling reactions were performed as in A, B and C additionally to each reaction random oligonucleotide were added.