

**LSL** LOGIC SYSTEMS LABORATORY

**Laboratoire de systèmes logiques**



**Swiss Institute of Technology**

**Semester Project**

*Logic design for nanotechnology devices*

*Cellular Architectures Research Group*

*Author: Giorgio Brambilla*

*Assistant: Gianluca Tempesti*

## ***Introduction:***

Nanotechnology is one of the latest challenging fields of recent research. The miniaturization of the silicon circuitry is going towards the physical limit of the molecular structure. Achieving of this limit is the border of the actual technology.

There are physical constraints to the size of the computer components. Silicon has surpassed the Moore's law forecasting doubling of devices performance and density every one and half to two years. There are unavoidable obstacles involved in this continual downsizing of silicon and it can be seen that these limits will be reached in the next five to ten years. This leads to charge leakage and band structures becoming too small for proper function. Molecules have a relatively large energy level of separation also at room temperatures and nanometer-size scale.

“Over the past forty years, a total of  $10^{20}$  transistors have been made. But in a single drop of water there exist about  $10^{21}$  molecules” [nanocell logic gates]. Drawing from this relation it is realistic to demonstrate the possibilities of nanochip technology.

The recent milestones of this research direction have seen the development of new instrumentation. Subsequently, this has led to the opening of new frontiers to the exploration of the nano world realities and possibilities. Currently, various fields of nanoapplication research have been dedicated towards robotics, medicine and electronics.

Chemical and electronics collaborative research are mainly focused on the development of synthetic molecules with capabilities of assuming two or more different stable states. The growing of these molecules in a defined flat or three dimensional structure allows further interesting computing applications. The assumed state from each cell will be a consequence of the states of the neighboring cells according to a defined truth table.

These devices will permit the elaboration of “some kind” that are still not fully-defined. To date, we have not fully realized or understood the potential limits of these devices. In other words, we could not define and narrate the kind of elaboration processes that will be possible in the future on these supports.

# ***Index:***

## ***- Introduction -***

***Architecture Cellular array vs Von Neuman***

### ***Assumption***

*Failure detection and auto-reparation*

*Genome programming*

*Asynchronous behavior*

*Reversibility:*

### ***Research fields***

*Nanocell Logic Gates*

*Molecule Cascades*

*Asynchronous cellular array*

*Quantum-Dot Cellular Automata*

## ***- Demo -***

### ***Project Proposal***

#### ***Properties***

*Simulate asynchronous behavior*

*Cell matrix*

*Boundary conditions*

*Signal Path*

*Signal Propagation*

#### ***Implementation***

*QCA*

*Clock*

*Structure*

*Hand-made truth table*

*Self evolution*

### ***Conclusion***

### ***Bibliography***

## ***Architecture Cellular array vs. Von Neuman:***

The early days of the microprocessors of the *Von Neuman* architecture, as characterized by the separation of the memory and processing resources, has been leading the processor market. This architecture will be seen to pose severe challenges when superimposed onto nanotechnology [1, 2, 3, 4].

The requirement for an irregular structure and the lack of real wire between different logic modules complicates cost-effective manufacturing on a molecular scale and thus pose problems of global connectivity. At higher integration densities, computer based on the *Von Neuman* architecture are not gate limited but rather interconnection limited [3].

Architecture based on cellular arrays suffers less of such problems. These are locally connected in homogeneous patterns that contain a vast number of identical cells.

## ***Assumption:***

It is clear that there exist close relationships between nanodevices and the cellular automata theory. Some important behaviors are provocative and speculative resulting in their exploitation as starting points for previewing of possible future employments.

### ***Failure detection and auto-reparation:***

The knowledge and control of nanochip synthesis does not guarantee a compatible regular structure and can be attributed to an unavoidable percentage of faulty molecules. The growing of the desired structure of molecules will lead to different possible fault:

1. Wrong positioning of the cell in the desired structure. This leads to:
  - erroneous interaction with the neighbours;
  - wrong orientation of the cell owing to a lack of symmetrical alignment of the molecules.
  - Void spaces;
2. Fixed cells occur as a spontaneous ability of faulty cells to attain a functional state on the basis of an accurate building process.
3. Unpredictable behavioral errors occur as a consequence of surrounding environmental noise such as electromagnetic interaction, resulting in certain unpredictable changes of state.

A theory in the management of permanent or temporary defects is much warranted. The capability of managing defects corresponds to these behaviors:

- Failure detection
- Auto-reparation or deactivation of an area, in order to avoid error propagation.

All these features are already well developed in the bio-inspired cellular automata theory.

### Genome programming:

Adherence to a silicon board serves as the most desirable way of wiring these devices. The silicon micro layer serves to provide adaptation and amplification of signals to the nanochip. This base layer will also provide the power supply and the cooling system.

The ability of the nanochip to adhere on a solid state semiconductors board relies solely on scanty wired paths on the sides of the structure for possible interactions. This should drive us to think towards a “black box” system, analyzed only by the border behavior.

These constraints force the programming to two ways:

- Static programming must be provided at the construction level. It can be achieved by fixing the state of some molecules [demo: “Hand made truth table”] [5].

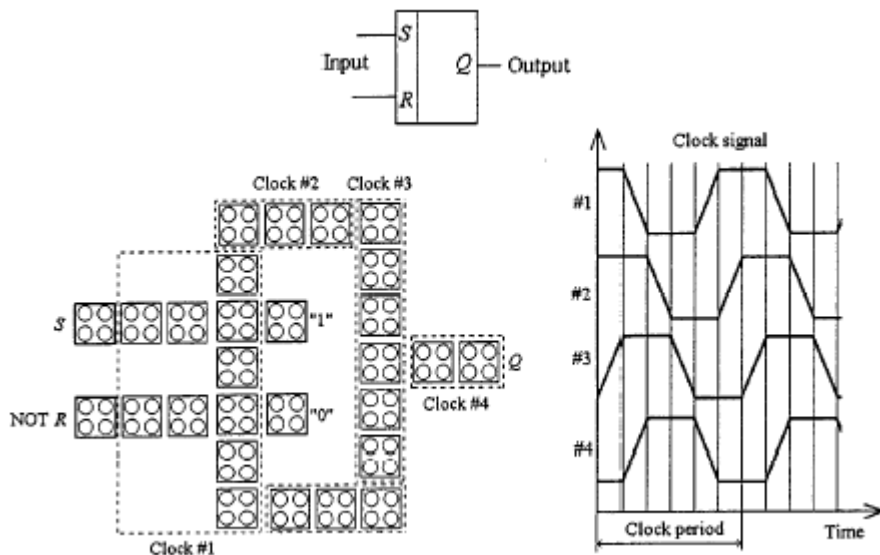


Figure 1 - Single-bit memory: SR flip-flop.

- Dynamic programming (nanocell logic, demo1) is performed by insertion of a sequence of information by the wires on the sides. This lead directly to the genetic programming theory, applied to a matrix of molecules. The application of the genetic programming theory, different genome inserted in the wires. This programming must follow a set of constraints:
  - o Each cell has only two or few stable states;
  - o Each cell does not possess position notion. This information is commonly used to select the corresponding genomic sequence.

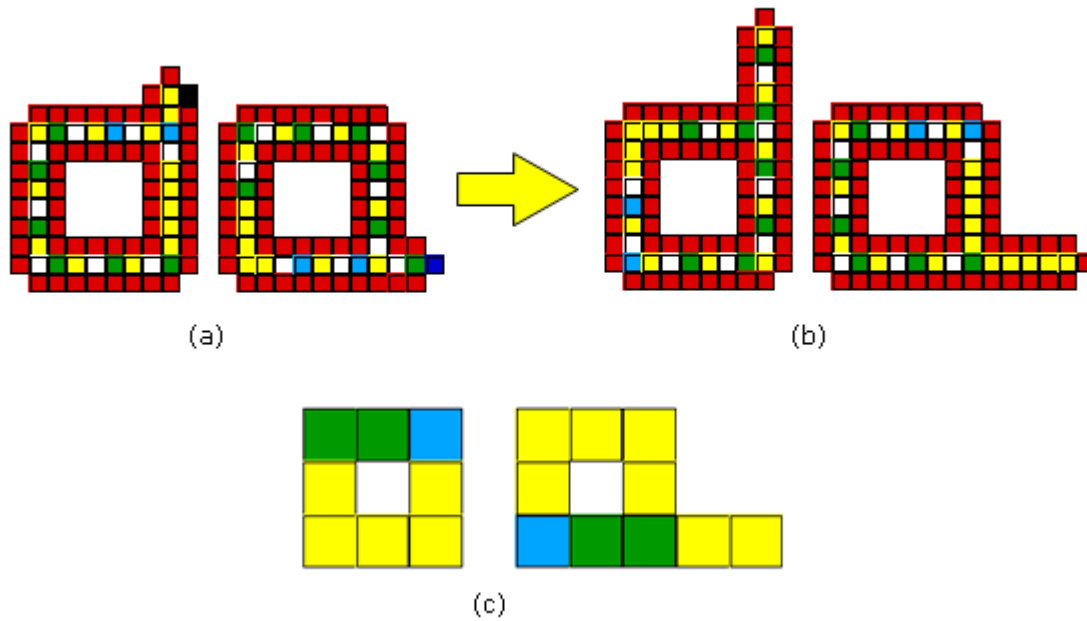


Figure 2 - (a)(b) Langton's Loop Construction (c) Reggia's Loop.

### *Asynchronous behavior:*

Almost all digital devices on the market work in a synchronous fashion. The cellular automata function in the same way, the updating of their state at each time point follows truth table rules.

It is improper to speak off synchronous behavior in nanochip. There are more than one physical obstacle:

- It is prudent that the clock reaches every cell; in the case of bi-dimensional structure the matrix must be completely wired.
- The signals must arrive at the same time to the whole chip. The switching time for a nano cell will be less then the propagation time of the clock.
- The arrival of the clock to every molecule imposes a specific input for the signal.

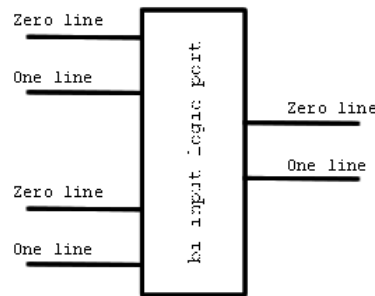
In an asynchronous computational model all cells conduct simple operations timed randomly and independently of each other. Though attractive for attaining efficient physical realization, this system brings to mind the question of computation on such cell array.

In order to accomplish the asynchronous constraint, a manage dataflow system must be provided. The biggest problem for this kind of asynchronous systems is the management of the dataflow on the data line. It is mandatory that the signal is holding sufficient time for the achievement of all the required operations.

The few attempts [6, 7, 8, 9] have focused on simulating a timing mechanism on an asynchronous system to force cell into synchronicity, for later utilization on well-established method to compute synchronously. This results in inefficiency: 1. timing mechanism requires increased complexity of the cells; 2. this method do not permit the exploitation of the massive parallelism of the cell array.

Other strategies to achieve asynchronous management:

- Retroaction:  
It is the function of a lot of modern asynchronous systems. Different modules are spread on a device. Upon the completion of each elaboration step an acknowledge signal is sent.
- Signal doubling:  
This system does not permit a real synchronization. It consist of using two different “data line” for each bit of information. One is the zero line and the other the one line.



**Figure 3 – Example of unary logic component.**

- Timing:  
In a combinatorial circuitry we can preview the maximum time for reaching all the components, dependent on the speed and dimension. This saves on complexity but not optimization of performance.
- Delay intensive circuitry  
This is the most efficient and simple system. A type of circuitry that allows arbitrary signal delays without being an obstacle to its correct operation. The only drawback is the quantity of required state. It results in difficulty in their implementation on a bi-stable cells system. [10 11 12]

All the method described requires particular behavior by the system. As the retroaction requires the use of data line in an opposite direction as the usual dataflow; this behavior could be a problem if the molecules will have only few stable states.

### ***Reversibility:***

Reversible computing is virtually unexplored in an asynchronous framework. As most physical interactions on the nanometer scale are asynchronous, it makes sense to investigate whether a reversible mode of computation can be included in an asynchronous framework.

Reversible device combined with dynamic genetic programming permits the achievement of a vast array of varied computational task..

## ***Research fields:***

At this point we have defined “at row line” the architecture, not based on a *Von Neuman* machine in order to maximize the computation parallelism. This is followed by the definition of the main constraints that this architecture must follow. Recent research efforts have been grouped into various order which tries to follow different strategies toward the nanochip target. The greatest challenge lies in the merging of advanced computing knowledge and molecular chemistry. Below summarizes some of the results achieved by different research groups:

### ***- Nanocell Logic Gates:***

#### ***Reference:***

- “Nanocell Logic Gates for Molecular Computing”  
IEEE Transaction on Nanotechnology, Vol.1 No.2, June 2002
- “Programming the Nanocell, a Random Array of Molecules”  
Rice Univ, Houston, TX – Ph.D. thesis Summer M. Husband

#### ***Research by:***

- Rice University, Houston, TX – USA
- N.C. State University, Raleigh, NC - USA

#### ***Brief analysis:***

This research group has proposed and implemented successfully a chip based on a silicon board. The connections are based on a group of paths on the side of the nano chip. Their working model is a fairly regular hexagonal nanochip possessing a thin surface of nanomolecules, connected in a random way. Each cell possesses connection range from 0 to 6, according to a Poisson distribution average 5, to his neighborhood.

This network of molecules and particles can be modeled as a planner graph where the nodes are the nanoparticles and the edge are the molecular switches.

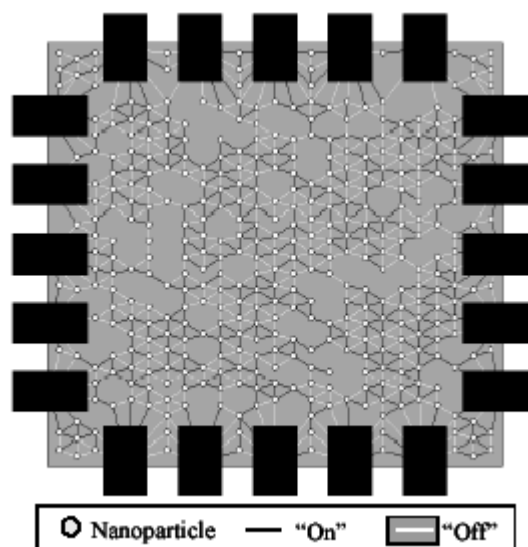
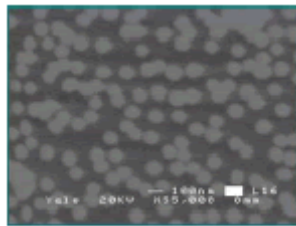
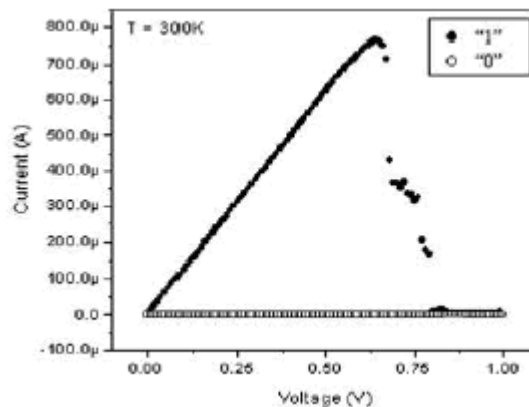


Figure 4 - This is a simulated nanocell with five I/O leads on each side.



**Figure 5 - The deposition of gold nanoparticles in a physical nanocell is displayed.**



**Figure 6 - This is the IV curve used in simulating nanocells. The “on” to “off” ratio is 1000:1. Such behavior has only been obtained experimentally at very low temperatures (approximately 60 K), but chemists on the nanocell project expect to see it at room temperature in the near future.**

This regular grid in the simulations has no effect on the programming of a nanocell. It only determines the possible connections within the nanocell.

Once the physical topology of the self assembly is formed in the nanocell, it remains static [...] The only changeable behavior is in the molecular state: conducting ON or nonconducting OFF.

Several types of room temperature operable particles have been synthesized and successfully demonstrated [13, 15, 16]. The connections between the cells are the active parts of the circuitry; their behavior follows a simple Voltage Current function.

*Behavior:*

Switching will occur only through voltage pulses between the external path. Programming involved the knowledge of the switching graph and the correct connection path. For usage of these circuitry it is mandatory to find the correct switch state for performing the target logic device and the correct series of voltage pulses applied to the I/O pad. This sequence must rise or lower the state of the identified molecules toward the achievement of the target function.

As the structure of the internal connection is randomly distributed, a search function must be applied to the circuitry to find configurations. During simulation genetic algorithms have given good results in comparison with other heuristics. In this case a chromosome corresponds to a different set of switch state for the nanocell with a fixed location of nanoparticles and molecular switches.

*Defect and fault tolerance:* It has been demonstrated that this kind of nanocell have a good effect tolerance.

Performing of a simple logical operation in this architecture can involve the participation of several switches. It indicates that a nanocell trained as a logic function is defect/fault tolerance. The robustness of this system is observed in its high tolerance to a degree of faulty switches [14].

***Comments and questions:***

This field of research has been pursued only by a few research teams and there has been no recent report on further progress.

The perspectives for these kinds of circuitries do not go further than a simple kind of combinatorial function. In other words, the result achieved by these techniques does not see a real challenge for commercial applications that involves complex processor computation. Meanwhile, these systems serve as working prototypes that have already produced concrete experimental result.

***- Molecule Cascades:***

***Reference:***

- Molecular Cascades, Science 298, 1381-7

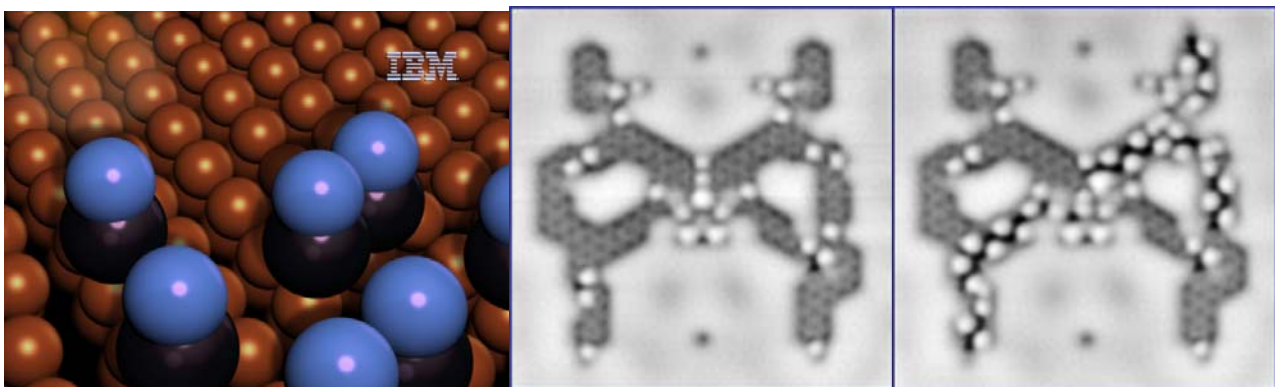
***Research by:***

- IBM Nanotech laboratory

***Brief analysis:***

One of the only experimental success about the molecular computing was performed a few months ago by the IBM facility in nano application research.

The technique known as “molecular cascading” permits the achieving of logic gate and molecular wire using CO molecules on a Cu(111) surface.



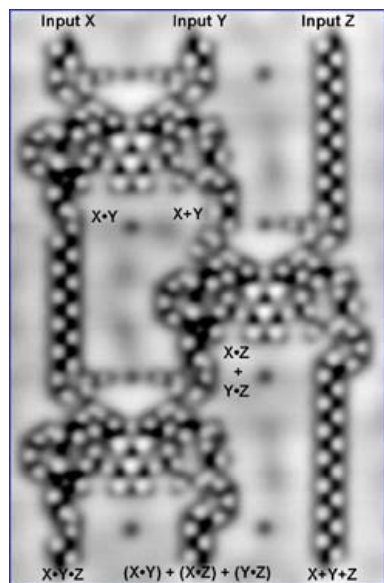
**Image 7-1 – 3D simulation  
7-2 & 3 – Two-Input Sorter**

The group has claimed that the molecule cascade serves as a novel approach to perform computation, and this exist as a premier construction for all of the components necessary for nanoscale computation, connected and then made to compute. It is also the smallest existing operating circuits ever manufactured made to date.

***Comments and questions:***

As this system originates from private facility research center, the propriety rights of this system will not be rendered it easily accessible to the public. The IBM experiment has given great result in this field.

The huge drawback is seen in the irreversibility of this kind of circuitry. In other words, after a computation the entire circuitry is unable to perform further computational tasks. And it is easy to imagine a chip that works only once do not have big possibilities of flooding the computational market!



**Image 8 – Three input sorter**

**- Asynchronous cellular array:**

***Reference:***

- Molecular Cascades, Science 298, 1381-7

***Research by:***

- Communication Research Laboratory, Nanotechnology Group, Kobe, Japan

***Brief analysis:***

As one of the most advanced player in the area of asynchronous system of cellular array, the research facility represents a tie between chemists and computer science groups. Their biggest effort lies in the direction of the delay intensive circuitry. Their approach was based on a cellular array architecture that conducts randomly timing operations requiring four bits of memory for each cell. *Delay insensitive circuits* may lead to an increase in the performance in the system that takes advantage of the massive parallelism permitted by this architecture, e.g. simulation of particle systems in physics, neural network and biological simulation.

symbol						
state	0	1	2	3	4	5

Figure 9 - The symbols by which the cell states are encoded.

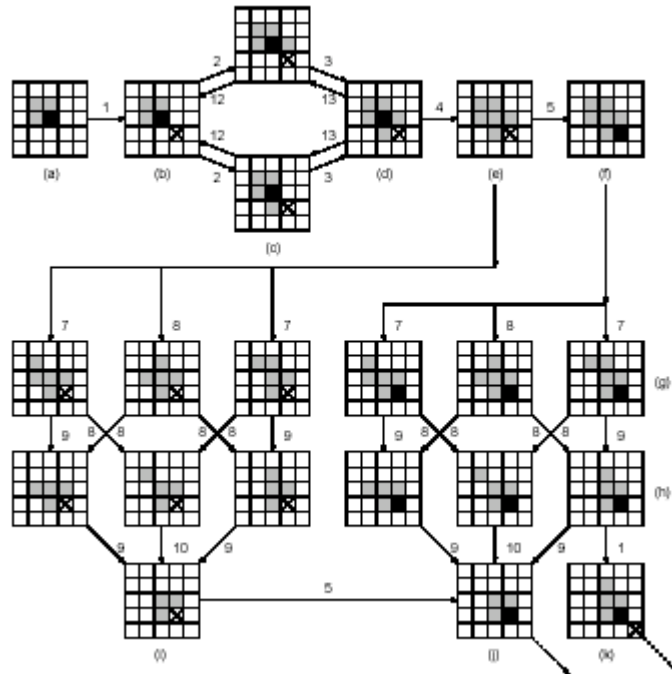


Figure 10 - Propagation of a signal on the asynchronously updated CA.

The requirements to achieve universal computation in cellular array are largely different to those of solid state semiconductor electronics. Wire and specific connections possesses no sense in this nanostructure. Information flow-through does not traverse solid cable in the form of a current, but as a chain of action-reaction of neighboring molecules.

The operation required to achieve universal computation in cellular array are:

- 1 – signal propagation, in at least two direction;
- 2 – data split and join rules;
- 3 – combinatorial logic port (as in the silicon architecture).

This goal is achieved using an entire set of primitives based on the *Von Neumann* or *Moore* neighborhood for exploit universal operations.

One of the most interesting challenge in the field has been about the configuration of this system. The two way that are prospected in the early future could be summarized as follows:

- 1) *Computational* and *Configuration* modes. Computation perform all the operation needed for a universal computing, configuration is subsequently achieved by rewriting the memories from cell to cell until the final destination.
- 2) *Self-reproduction* This second method utilizes the collective behavior of the cells to copy information from one part of the array to another. This is more challenging and opens a series of interesting applications if one perceives it as a *self-reproduction* method. There are various reports on the success of these tasks in the synchronous cellular arrays [17, 18, 19, 20]

The other undeveloped field is the defect and fault tolerance. In principle, defect and fault tolerance are possible in asynchronous cellular arrays, but necessitates further research to design models implementing this concepts, without further increasing the complexities of the cell.

***Comments and questions:***

The last month have seen a number of interesting reports made by this group. It is interesting and certainly encouraging to note that their efforts have been specialized on many of the different aspects of involvement in a cell array computing, based on asynchronous systems, e.g. programming, signals propagations and fault tolerance.

A lot of topics that involve these research areas have yet to be exploited and represents future challenges, as for the reversible computing or the configuration problems.

Topics for future research may be as follow:

- 1 – Reversibility
- 2 – Defect tolerance
- 3 – Physical implementations of four bit cells

A lot of simulation are elaborated and tested by this research lab, but a drawback lies in that any physical implementation that support their “ideas” have yet to be built and tested.

***- Quantum-Dot Cellular Automata:***

***Reference:***

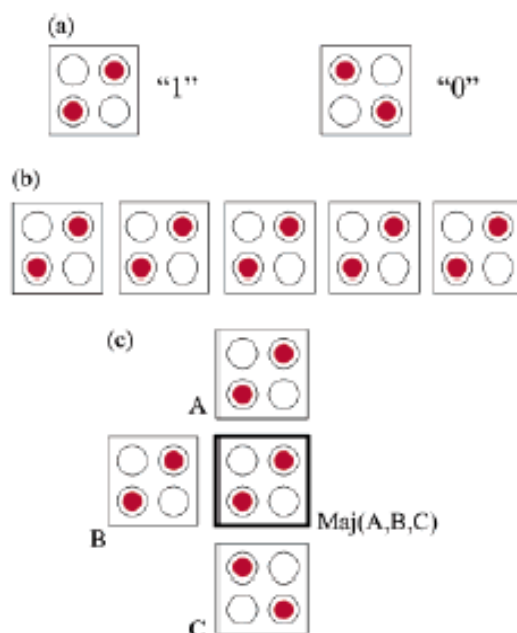
- Signal processing with Near Neighbor-Coupled Quantum-Dot Arrays  
Craig S. Lent, N.D. Univ., IN - <http://www.nd.edu/~qcahome/>

***Research by:***

- Department of Electrical Engineering, University of Notre Dame, Indiana USA

***Brief analysis:***

Quantum Dot Cellular Automata (QCA) represents one of the most attractive compromises between all projects under development. This project follows mainly the guidelines defined in the previous chapter focusing on asynchronous behavior. Moreover, there are no current reports studies about fault detection or defect tolerance. The configuration process of the cellular array, to improve specific function, is also not involved as research field.



**Figure 11 - Schematic of QCA cells. (a) states 1 and 0; (b) chain of signal propagation; (c) Neighbor interaction**

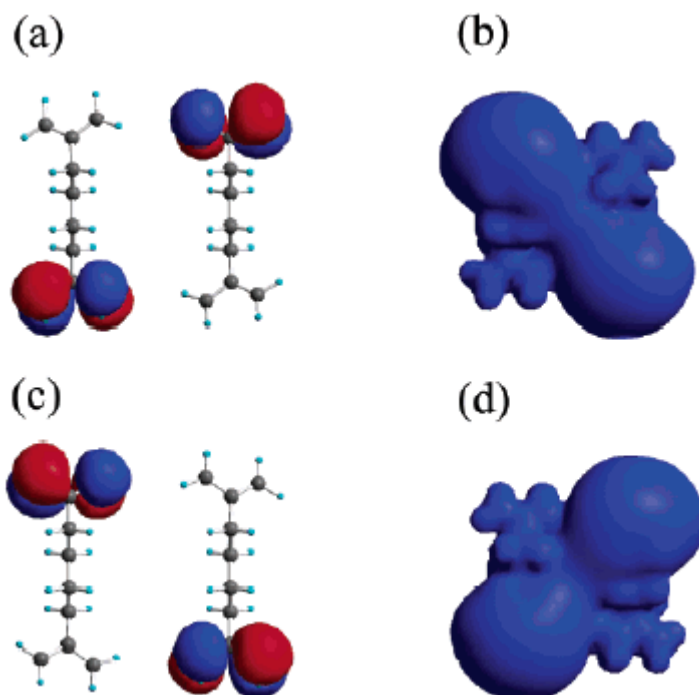
The architectural proposal is a QCA molecule cellular array. Here, each molecule can assume only two states, corresponding to the binary values of one and zero. If a cell is not excited by an external influence it assumes a neutral state and does not interact. The excitation of a cell arrives as a consequence of a proper neighborhoods combination.

A truth table (table) is to be proposed and used as a base to demonstrate the feasibility of many computational processes. The evolutionary rules for the cells are based on a *Von Neumann* neighborhood. The lack of the involvement of the previous memory state on the next state computing process means that any memory skills are not directly possible using this configuration. Nevertheless, a memory based behavior which is possible through sequential operations, can be obtained as demonstrated [memory cell], by building a complex structure that involves a loop of cells to maintain a specific state.

These characteristics resulted in the easy implementation of this kind of molecule. The synthesis processes is less cumbersome for such a molecule with only two stable states. The technological and manufacturing skills required for such molecule are surely less than the once described in last two proposal.

QCA True table	
N, E, S	Q'
0, 0, 0	0
0, 0, 1	0
0, 1, 0	0
0, 1, 1	1
1, 0, 0	0
1, 0, 1	1
1, 1, 0	1
1, 1, 1	1

**Figure 12 – QCA proposed truth table.**  
 N = north, E = East, S = South, Q' next cell state, in figure 11 are called A=N,B=E,C=S.



**Figure 13 - The HOMOs of the two stable states of the 2 dication are shown in (a) and (c). The corresponding isopotential surfaces ( $|V|=5au$ ) are shown in (b) and (d).**

Various studies demonstrated feasible operations employing the QCA architecture like simple logic port and combinatorial elements. Although it appears that a QCA system can perform universal computation, the constraints applied have rendered the results largely unimpressive.

The dataflow is propagated in the direction defined by the alignment of the molecules. This implies that the construction process must control the orientation of each cell in the array. The underlying perspectives in nano manufacturing technology will be narrated by uncertainties of the orientation and structure of the nano-molecules.

Further consequence arising from the absence of circuitry re-configuration after the building process, renders the auto-repair difficult and expensive.

The placements of the cell in the array are irregular. This avoids a possible growing of the matrix as a crystal in the building process and necessitates a building process that controls exactly cell placing.

A fixed circuitry boundary prevents any wrong interaction with cells that are not directly involved in the process.

Finally it is yet unclear: 1- how the activation level could be propagated in the array. 2- how the circuitry could choose the cells to activate, 3- the critical energy requirements after a chain of operations.

***Comments and questions:***

The QCA represents the most simple and short term realizable molecule which lends particular interest to this study. On the other hand, the extreme difficulties of placing the cells in the correct structure render many of these studies largely unemployable. The architecture proposed do not maximize the use of the distributed computation rendered possible by the cell array structure.

Fault detection and auto-repair management, essential in a nanochip implementation.

Largely this system still lacks the essential properties of fault detection and auto-repair, owing to the absence of dynamic programming at the molecular layer. This makes nanochips a less attractive competitor than the solid state semiconductor chips.

## ***Demo:***

### ***Project Proposal:***

After a preliminary analysis we have chosen to produce a demo in order to simulate the discussed topics.

We have rank the analyzed proposal using different parameters with the feasibility of achieving complex computation. The QCA architecture is the model of best feet to the defined parameters.

QCA does not necessarily represent the most feasible structure, but a good probability of been implemented in the next years by various groups. It does not represent a most powerful system, as the *Array Asynchronous* proposal has proven to better. But a good representation lies in a compromise between the two different factors.

The proposal made by the Notre Dame University appeared to posses the biggest drawback with a high number of constraints for the system to be built and function. Our efforts have been to eliminate these constraints and looking to the proposed structure for possible modifications of computations utilizing the QCA molecule. We have maintained the asynchronous behavior as apply in the first chapter.

The elimination of some constraints has brought a lot of open challenges and an augmentation of the research space for possible solutions but yet guarantying a physical implementation in less time. The difficult challenge to be achieved in the QCA solution is the placement of each singular cell in the desired structure, and the definition of a specific orientation, that may be different between one cell to another. Also, the alignment between the different rows in the matrix structure is not guaranteed (figure). This lack of symmetry prevents possible growing of the matrix as a crystal mentioned earlier..

The demo is simulated in a uniform matrix. In this structure the orientation of the cells is the same for all. Nevertheless, it would not be possible for the elimination of an asymmetry in the behavior of a cell. This led to the fact that all cells possessing the same orientation is critical in the determination of the next state. In another sense, neighboring cells to the north or south face of a cell could generate different responses

The *Von Neumann* neighborhood is applied as in the QCA architecture, but our simulation have two additional characteristics: 1) the current state of the cell is applied in the truth table each having a memory thus permitting doubling of possible behaviors; 2), the truth table is dependent on the cell state on the right side., please refer to figure.....

In the demo we have applied a technique to achieve a dynamic configuration which was further complex in combination with the *Von Neumann* neighborhood and the asynchronous propagation

The defect and fault detection and repair represent an ongoing challenge. But one of the simulation executed successfully demonstrated that it can adsorb a number of fault or defects.

## ***Properties:***

### ***Simulate asynchronous behavior:***

During the demo design phase, it has been difficult to realize the simulation of a real asynchronous system. On the other hand without a more realistic simulation the asynchronous behavior must be implemented in order to give a certain trueness to the results achieved.

*Java* is the chosen programming language; the simulation is performed on a common PC which works in a strictly synchronous and sequential way. The demo circumvents the constraints due by the sequentially execution of the process in a common PC.

The only way to facilitate the simulation of a parallel massive computational is to use different threads in the same process.

The other challenge lies in the recreation of “randomly timing activation” as in the asynchronous system.

The structure resulting in this way a matrix of cell

This property lies the structure towards a matrix cell where each cell correspond to a thread. So that the state of the cells can evolve independently.

Operating system management policy lies toward a sequential activation of the threads. This leads to a constant thread activation sequence, avoiding randomly timing activation. A random thread activation sequence is provided manually in the demo to circumvent any sequential execution imposed by the O.S. timesharing algorithm.

### ***Cell matrix:***

Upon the definition of the asynchronous behave of the matrix structure we have to define how to define the accomplishment. The best way to simulate the cells is to create an object that represent the single molecule and has the same properties. This techniques permits application of a thread to each cell and analyzed the state evolution process.

The way as *Java* manage the access to the class properties permits the random activation of each molecule described before. For instance the request for the state of the neighbors' cells involves the access to other cells without interfering their behavior.

*Java* methods, if not specify as synchronous, can be accessed and executed at the same time by different threads. The programmer than has to guarantee the data consistency of the object accessed, in our case all the concurrent access are made only for reading proposal, this do not involve consistency problems.

### ***Boundary conditions:***

As the evolving of the molecule state depends from the state of the other cell in a matrix, we must define the boundary condition of that matrix. The more expected nanochip physical implementation will have constant and uniform boundary values. Constant side for the building process “sounds not so difficult”.

In the demo all the border have value fixed to zero. This does not involve that state “one” is not expected as possible boundary state. The situation is actually symmetrical and can be the same if the truth table is completely inverted. This means that the value “zero” and “one” do not have any meaning “a priori”, and do not involve any specific process. Is just a random choice that does not render any adding constraints to the problem except for the uniformity of the boundary.

### ***Signal Path:***

At this time we have to specify how to accomplish external path to the nanochip. How put in communication the nano-structure and the rest of the world are still not clear. We have suggested a plausible solution.

Input and Output connection can be analyzed in two different ways. The output path are not directly visible in the demo but are represented by the user interface. The user can evaluate the value of all the cells. In the real implementation it is expected that some cells, especially those placed on the border of the structure, would be able to communicate their state to the external world. Or as in the demo we expect possible the external observation of the change in a particular molecule.

The input cells are expected as cells whose state can be settled externally of the nano-structure. A sequence of clicks applied to the user interface lead toward the complete configuration of the chip, then data-flow input follows using the same way. In any simulation only the boundary cells has been used to transmits dataflow, as expected in a real implementation of the nanochip.

### ***Signal Propagation:***

As seen before the signal propagation represents an interesting challenge. It dependent on the timing-regulation algorithm chosen (see chapter “Asynchronous behaviors”). *Timing* is the chosen strategy in the signal propagation management of the demo architecture. Simplicity in programming has been a criteria for its choice, other strategies involved auxiliary circuitry to be managed. For example the *signal doubling* require a cross management module in order to avoid line crash, on the other side *timing* do not maximize the circuitry performance which is not the target of the demo. The difficulties in the application of more efficient strategies are the reduced dimension of the truth table due to the reduced number of neighbors considered. The possibility of accomplishing those depends on the engineering efforts in the near future. This can be seen in the retroaction of an acknowledge signal often used in the asynchronous system involving signals grown in opposite direction, hence resulting in difficult management.

*Timing* waits for signal propagation towards all the possible gates. An estimation of the propagation time is possible through the elimination of data loops. The propagation sequence of the concurrent signals is not important in the combinatorial scheme, and in the case of sequential component (as latch in the demo) the control is due by forced delay sequences.

## Implementation:

Four different programs based on the same skeleton have been developed, each one finalized to demonstrate different behaviors. The development focus on finding advantages and drawbacks for the strategies involved.

The five demo are characterized by the properties above with each one focused to a specific problem.

Summary of implementations:

- QCA
- Clock
- Structure
- Hand made truth table
- Self evolved

Follow the brief description of each program, the result achieved and possible application scenario.

### QCA:

The initial simulation focused on the usefulness of the Lent [chapter before] truth table. The behavior of this cell is strictly directional (see image). The approach lies in the analysis of an ordered matrix of QCA cells ability to perform useful computational tasks.

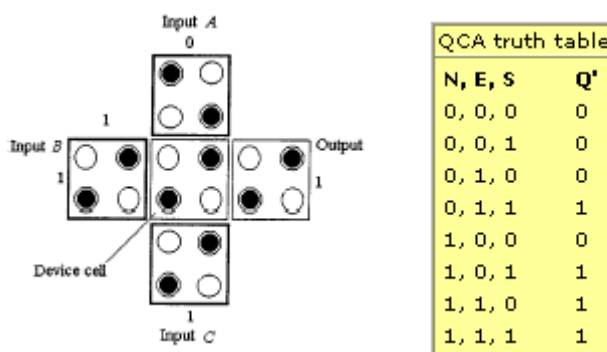


Figure 1, 2 - Majority logic gate.

The truth table proposal is based on the simple alignment rules of the QCA. Each cell takes the value assumed by the majority of the neighbors.

This rule appear deterministic to the paper analysis, the state are consequence of only 3 neighbors. This behavior becomes problematic when the state depends on 4 neighbors. This leads to parity problems that create indeterminate states. A molecule based on this truth table in the case of two neighbors with value “one” and two with value “zero” can assume an in-deterministic value.

The more likely behavior is the assumption of a memory state being a consequence of the sum of influence that is null. The energy necessary for a state change may be unattainable as a random consequence of environmental noise. Anyway the *memory-state* is only a supposition; the *parity problem* was not mentioned in various reports about QCA arrays.

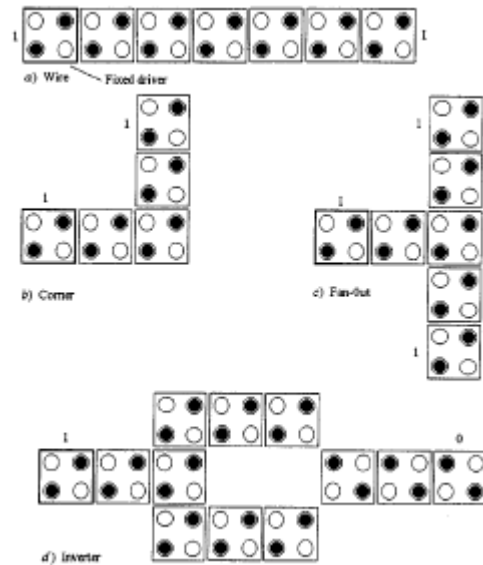


Figure 3 – Example of QCA interaction

### Simulation:

The parity problem is managed in two ways in this simulation.

- 1) The creation of an error-state assumed by the molecule when parity happens. The target of this test is to achieve computation without reaching the parity state. The parity problem in the picture is indicated by the red color.

The simulation has demonstrated that interesting results can be achieved via avoiding of the parity-state. When the circuitry is stressed to perform some kind of applications it often happens that the red state is accomplished. Once the parity-state happens it undergoes rapid propagation thus covering the whole matrix. The parity-state creates indetermination which erases any other operation in progress in the same chip area. Only in a forced state boundary that results in blockage of the erasing process.

- 2) In this method the parity-state is considered to be a stable memory state. When a cell is in a parity state, its value will be maintained as before the parity change. Once the in-deterministic problem is solved, the studies are focused towards the accomplishing of interesting computation process and configuration. Different test cases were applied to determine a useful behavior, without “exciting” results.

It is difficult to achieve signal propagation within a uniform matrix. Once a chain process that propagates a signal is activated, this is in undefined directions towards the filling of all available computational space.

### Results:

In the absence of straight constraints imposed in the publication, this cell model do not permits the arrival of any interesting computational result. The uniform matrix is obviously an unfit for the model and the truth table proposed by the Notre Dame University research team.

**Clock:**

The discussion about the asynchronous system has identifies different strategies to manage the data traffic. At simulation level appear important demonstrate the feasibility of these models.

The effort concentrated on the reproduction of a *retroaction*, as a way to perform synchronization, by acknowledge signal has demonstrate the complexity of this solution. A series of primitives useful to accomplish have complex design. The same problem involves the realization of *signal doubling* system. And a delay intensive politics is certainly improper in a bi-stable structure as a QCA matrix.

The attempts have focused on the creation of a bi-stable oscillator that can reproduce an external clock but well integrated in the circuitry and working at a nanotech speed. The ideas is the creation of many of this clock and integrate one of them in each module. The level of parallelization of the computation in that case is directly connected to the number of nano-clock involved.

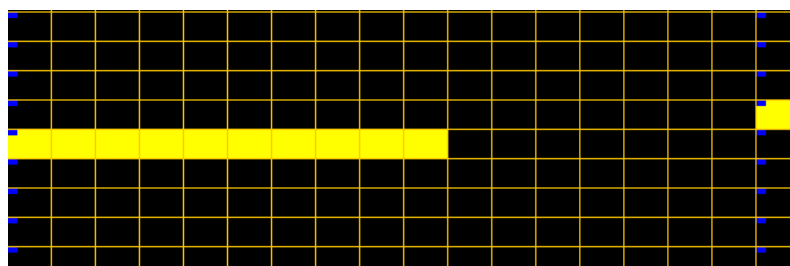
In the schema are reported the simple truth table needed to involve an oscillator behavior.

Old state = 0		Old state = 1	
N, E, S, W, Q	Q' Forced states	N, E, S, W, Q	Q' Forced states
0, 0, 0, 0, 0	0 Maintain stability	0, 0, 0, 0, 1	0 Line propagation
0, 0, 0, 1, 0	0	0, 0, 0, 1, 1	0 Line propagation
0, 0, 1, 0, 0	0	0, 0, 1, 0, 1	0
0, 0, 1, 1, 0	1 Clock	0, 0, 1, 1, 1	0
0, 1, 0, 0, 0	1	0, 1, 0, 0, 1	0 Line propagation
0, 1, 0, 1, 0	0	0, 1, 0, 1, 1	0 Line propagation
0, 1, 1, 0, 0	0	0, 1, 1, 0, 1	0
0, 1, 1, 1, 0	0	0, 1, 1, 1, 1	0
1, 0, 0, 0, 0	0	1, 0, 0, 0, 1	0
1, 0, 0, 1, 0	0	1, 0, 0, 1, 1	0
1, 0, 1, 0, 0	0	1, 0, 1, 0, 1	0
1, 0, 1, 1, 0	0	1, 0, 1, 1, 1	0
1, 1, 0, 0, 0	0	1, 1, 0, 0, 1	0 Clock
1, 1, 0, 0, 0	0	1, 1, 0, 0, 1	0
1, 1, 1, 0, 0	0	1, 1, 1, 0, 1	0
1, 1, 1, 1, 0	0	1, 1, 1, 1, 1	0

Figure 4 – Clock truth table.

***Simulation:***

The simulation has accomplished successfully the target proposed. Any rule guarantee that this kinds of circuitry work at a specific speed, but a minimum and a maximum propagation delay can be defined.



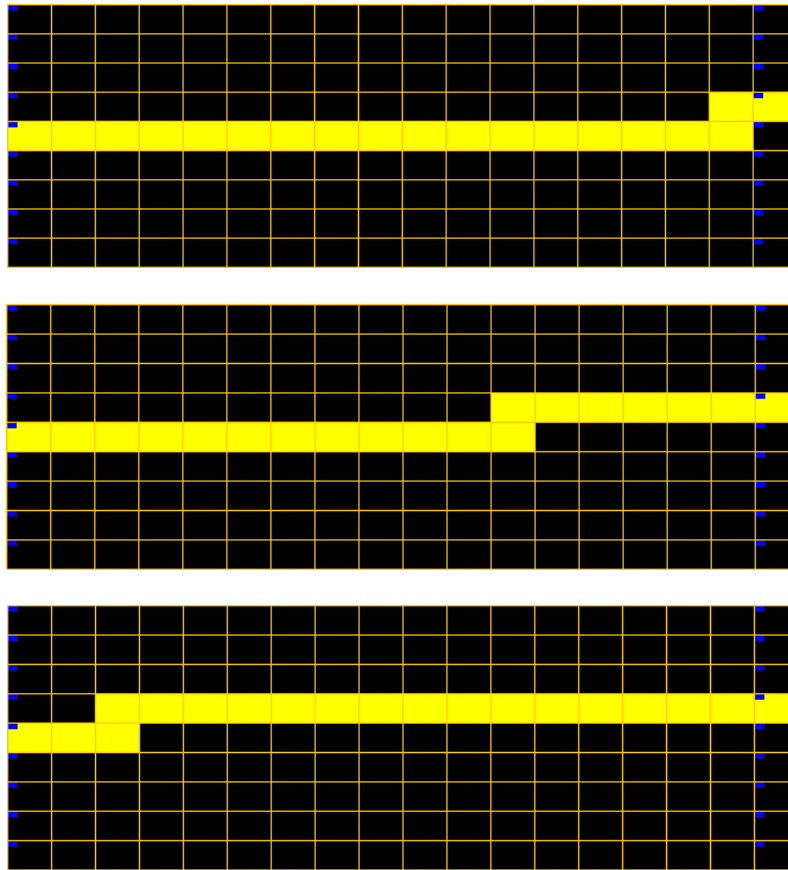


Figure 5, 6, 7, 8 – Sequence of clock signal evolution.

***Results:***

This program has demonstrated the possibility of the creation of clock and looping behavior in a QCA matrix architecture. This opens a wide horizon of possible applications for these circuits.

**Structure:**

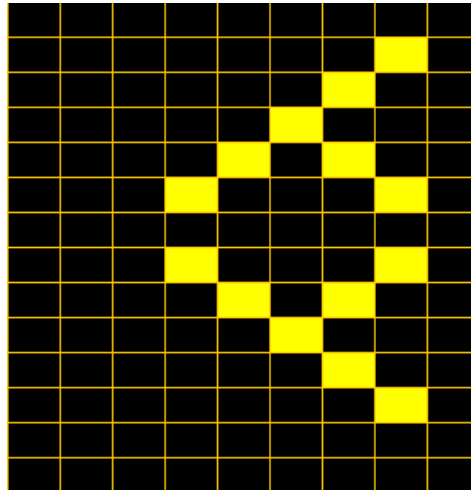
The study made on the cellular arrays has demonstrated that to perform dynamic configuration, the more attractive way of programming a nanochip, is useful the possibility of building structures. The studies in the bio-inspired field want to compare the grown structure with the molecular skin. In other hand share the matrix space in a series of structures with different computational behavior.

Comparing the cell array matrix implemented on silicon state with the one on the QCA arrays could be difficult. The cells used by the bio-inspired research are complex microprocessors with complex engineered interactions, not only bi-stable molecules. Nevertheless, the two implementations follow the same interaction rules.

The idea to demonstrate the possible comparison is to show the possibility of growing a structure comparable with the cell membrane. The biggest obstacle is the lack of the clock, the uncontrolled propagation of the signals in the asynchronous matrix is the biggest limit to the definition of a cellular membrane. The only structure that we have been able to grow with a specific truth table is

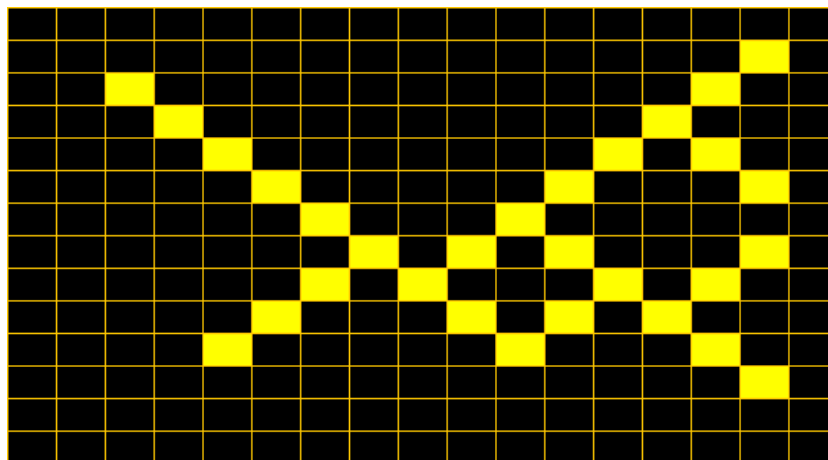
the ones show in this demo. Only a “stairs border” structure can grow using the Von Neuman neighborhood, for the result achieved in this project.

To achieve the dynamic configuration target the only rule defined is the once that permits the propagation with *Timing* strategy of the signals from the left to the right of the matrix.



**Figure 9 – simple structure dynamically programmed**

Once defined this rule we have elaborate a rule to erase the structure built, is enough create a certain local condition.



**Figure 10 – another example of structure dynamically grown.**

***Results:***

The possibility of creating a dynamic configuration rule has been demonstrated by this demo. This not only permits a dynamic structure but one with a regular or irregular cellular skin structure. Although encouraging targets has been achieved, the program demonstrated the un-usefulness of this structure at the current state of the art. Even though the structure may be rendered complete the success of the computational process remains to be seen.

**Hand-made truth table:**

The path towards the creation of useful functions has been resulting difficult in the pervious step; the effort was subsequently directed at finding a useful truth table.

The computational skills of an array of uniform bi-stable cells with all the other imposed constraints have yet to be determine, and currently there is the absence of a demonstration that proves the existence of at least an implementation as powerful as the Von Neuman machine. The demonstration is still a challenge of the *Information Theory*; thus only experimental result can be harnessed for making a demo for this project.

The target of this program is to search for a truth table that can be applied to a nanochip which serves to perform numerous computations. It is implied that during the building process the molecular value may be user defined in modular structure, before the computational phase.

*Data flow:*

Firstly, the management of all the dataflow was defined as follows: the left is chosen as input and the right as output thus providing the direction of the principal flow of data. A set of important rules were also applied in permitting this flow of data. The information go in the defined direction without following border or any other fixed cells.

*Vertical direction:*

In a complex circuitry, information arrival from different modules must converge in the desired component. In order to accomplish this task, a way is required in the transportation of data in transversal direction which corresponds to the demo in a vertical mode. The other set of rules that were free on the truth table were set to perform the *up* to *down* connections. This has been made possible by the usage of a fixed cell border enabling the data to follow in the desired direction.

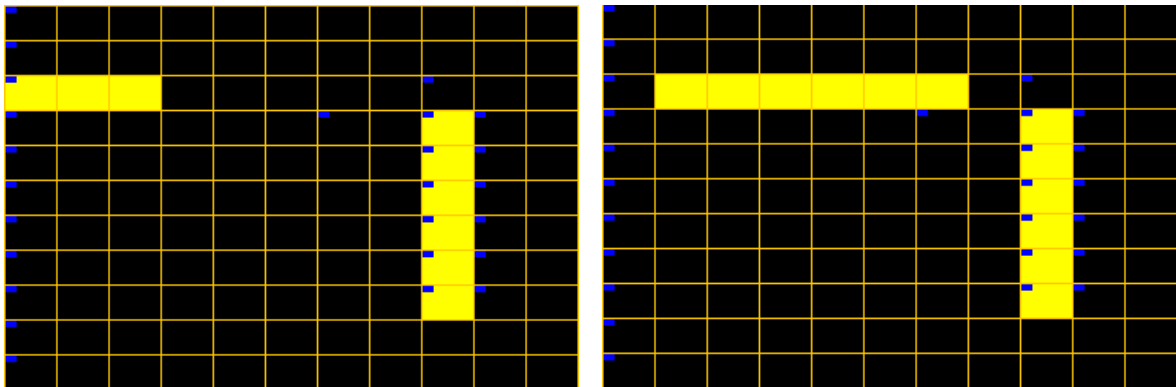


Figure 11, 12 – Sequence of “curve” propagation

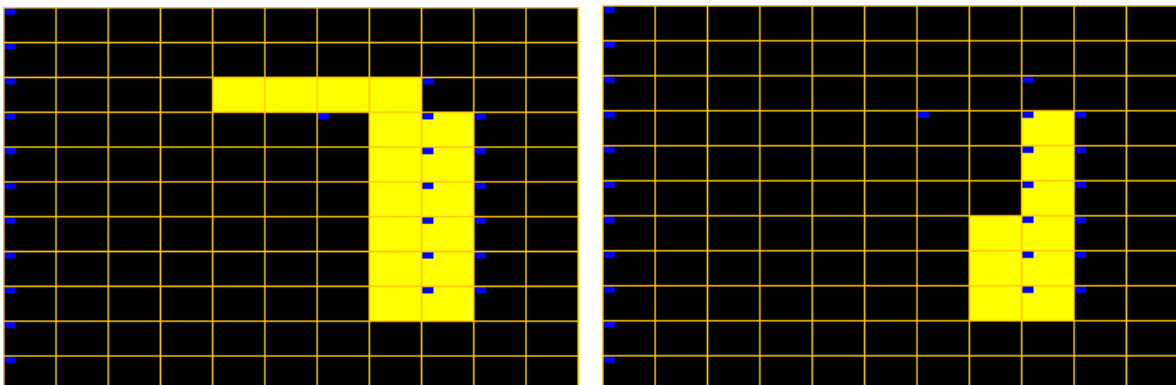


Figure 13, 14 – Sequence of “curve” propagation

The target of finding rules that perform on the same table with the data flow in the opposite direction was impossible in our analyses. This apparently limits the possible use of these circuits, avoiding essential functions for the universal computation. Example includes loops programming structure resulted impossible.

*Split function:*

The selection of the rules for the horizontal and vertical propagation has leads to the signal splitting as an important function to achieve the universal computation. As shown in the picture, whether a cell value is fixed or not, a module can assume curve or split signal function.

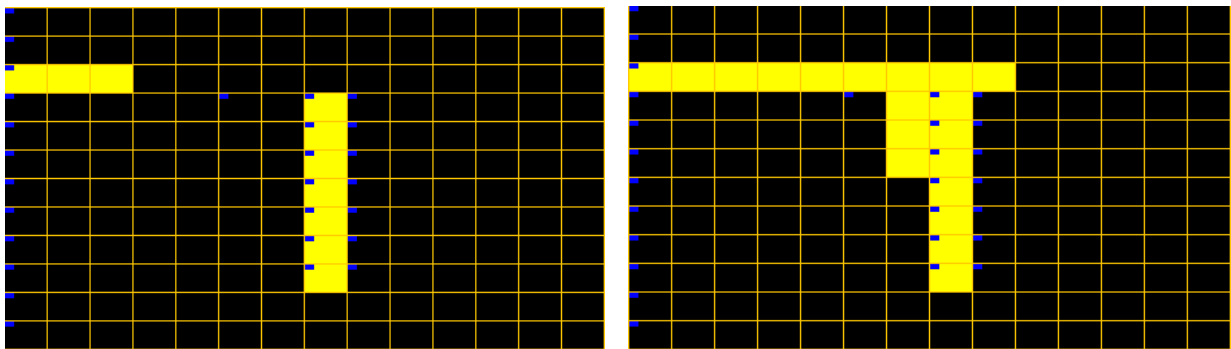


Figure 15, 16 – Sequence of “split” propagation

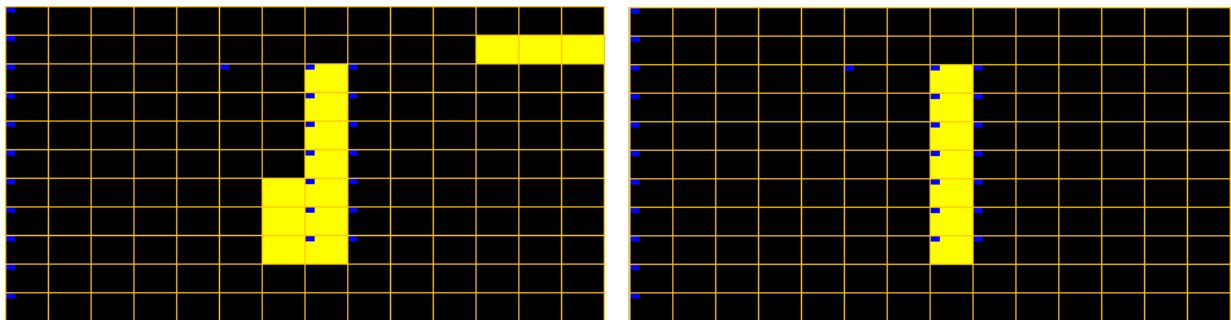


Figure 17, 18 – Sequence of “split” propagation

*Latch function:*

Once the basic data propagation structure has been defined, the matrix is able to lead data to the desired point. This leads to the focus on the accomplishment of a universal logic function. At this point of development, the truth table fixed rules are as much to represent a problem for finding interesting possible combinations. After futile efforts for a logic function employing NAND or a XOR, our main success has been on the identification of a latch function, that permits memory behavior.

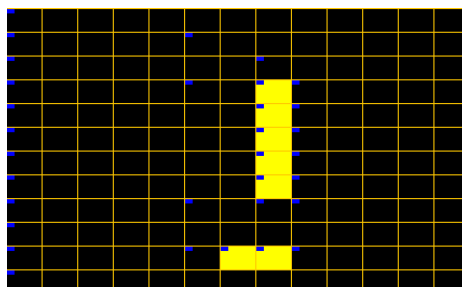


Figure 19 – Latch programming cell configuration  
(the blue square on top-left means fix cell)

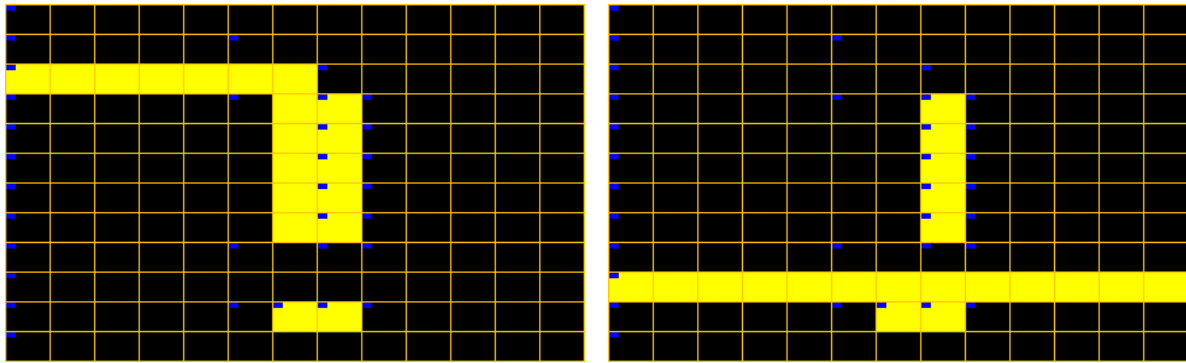


Figure 20, 21 – “rec state” active, “data line” transmit a signal

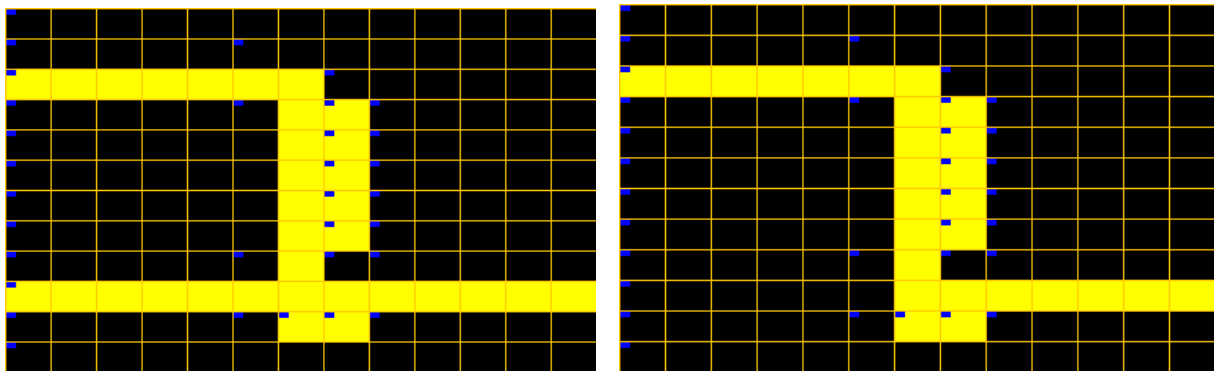


Figure 22, 23 – “rec the signal”, “hold” position, the signal is maintained after the input down

### **Results:**

Each function implemented on the truth table removes flexibility from the system. In this demo, we have been able to implement only these functions, the definition of a more complete set of functions require further engineering efforts. The path towards these four function, accomplished on the same table, demonstrate the interesting possibilities of the matrix molecule structure, but leaves many open challenges toward the way of the universal computation.

### **Self evolution:**

The effort in this phase is focused on searching a truth table that permits a universal computation. The artificial evolution permits the exploration of a bigger research space in comparison with the engineered solution. In the previous program, the engineering approach for finding interesting solution has implemented the desired function with a specified sequence and following an approach based on defined methods. For instance 1) strategy selection of data propagation, 2) method identification (e.g. Timing) ... Artificial evolution permits the identification of interesting solution, jumping of these constraints and evolving a randomly started system, dependent on the fitness function.

### **Implementation:**

Rules employed in the artificial evolution algorithm:

- Fitness roulette rule selection;
- Multi-point crossover;
- Elitism;

- Mutation.

The gene identifies the truth table and evolves on a fitness function that checks the correspondence with a logic function. The evolution focused on the enrichment of a XOR and NAND binary function on an array of 7 x 7 molecules.

*Evaluation step:*

The two input cells on the left are fixed to each combination (00 01 01 11) of the logic port for a defined time while the signal undergoes self-propagation in the matrix. After that time the system is stopped and the fitness function is calculated in relation to the value assumed by the selected output cell, on the right. If the value corresponds to that on the desired truth table, the fitness of the gene is increased by one.

Technical values chosen as evolutionary parameter:

- Population dimension: 100
- Number of generation: 20
- Genome dimension: 32bit (combination of the truth table)
- Percentage of mutation: 5 %
- Elitism: 4 individuals

The reproducibility of the results of the executed simulation has provided a set of good solution founded on the algorithm starting from the initiating three generations of the population. But this has resulted in instable solutions. A set of solution was discovered in the first step of the evolution, starting from a random population involving an intrinsic simplicity of the problem and the possibility of many different solutions. On the other hand, all the individuals with a maximum fitness analyzed have given an unstable solution; value measured for the fitness was maintained by the molecule targeted only for a few instants before changing.

The solution derived were un-useful for the identification of solutions because the output value is arbitrary between zero and one, the output cell appeared to blink in the demo user interface.

The solution to the evolution problems was often achieved via attenuation of the fitness function. The new fitness is measured by correspondence of the required output with the real output and also its stability. After a fixing time the simulation registers the value of the output cell and the fitness is increased if it maintained the same state for a quantum of time. Particularly interesting results have been arrived from another change of the fitness calculation. The quantum of time used for the stability measurement is increased with each generation, consequently at each generation the selected individuals are proportional to the stability.

***Results:***

The simulation of this program with the parameter above has taken an average generation time of 4-5 hours by a Pentium4 2.5GHz computer, and each simulation have provided an optimal 1 to 3 solution conforming to the desired logic port.

The only drawback is the lack of reversibility of the system which often after a series of port use causes blockage of the port.

A supplementary change of the fitness function may solve this reversibility problem by an increase of the testing combination for each individual.

## ***Conclusion:***

The research focuses in two main section, a first analysis involving the state-of-the-art of nanotechnology field in the computational area, a second part dealing in the assessment of the various reported concepts. We have also tried to identify some possible developments in future strategies.

The different demo has provided important and provocative results on the technique used to performing, on the method simulated and on the experimental side.

An important milestone achieved in the simulation has demonstrated that computations are possible employing a bi-stable molecule matrix. The universal computation on a nano-machine has never been demonstrated though many interesting sub-methods were feasible.

Many constraints and limits were identified, e.g. find a truth table behavior or exploitation of the problem due by the only two stable states that the chosen cell can attain.

Finding of a way to aggregate group of molecules serves as an interesting perspective; the effort to identify cluster of molecule leads to multiple stable states modules.

## ***Bibliography:***

- [1] Compańo R (ed) 2001 Technology Roadmap for Nanoelectronics (Luxembourg: Office for Official Publications of the European Communities)
- [2] Durbeck L J K and Macias N J 2001 The cell matrix: an architecture for nanocomputing *Nanotechnology* 12 217–30
- [3] Lyke J, Donohoe G and Karna S 2001 Reconfigurable cellular array architectures for molecular electronics Air Force Research Laboratory Report AFRL-VS-TR-2001-1039 available at [http://www-2.cs.cmu.edu/ phoenix/internal/papers by others/TR-2001-1039.PDF](http://www-2.cs.cmu.edu/ phoenix/internal/papers_by_others/TR-2001-1039.PDF)
- [4] Peper F 2000 Spatial computing on self-timed cellular automata Proc. 2nd Conf. on Unconventional Models of Computation, UMC'2K (Berlin: Springer) pp 202–14 - <http://www.cs.columbia.edu/ nowick/async-intro.html>
- [5] Csurgay AI, Porod W, Lent CS "Signal Processing with Near-Neighbor-Coupled Time-Varying Quantum-Dot Arrays" (Paper presented at PHASDOM'98, Neuchatel, Switzerland, 1998) *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I-FUNDAMENTAL THEORY AND APPLICATIONS* 47 (8): 1212-1223 AUG
- [6] Nakamura K 1974 Asynchronous cellular automata and their computational ability *Syst. Comput.—Controls* - 5 58–66
- [7] Peper F, Isokawa T, Kouda N and Matsui N 2002 Self-timed cellular automata and their computational ability *Future - Gener. Comput. Syst.* 18 893–904
- [8] Nehaniv C L 2002 Self-reproduction in asynchronous cellular automata Proc. NASA/DoD Conf. on Evolvable Hardware, EH'02 pp 201–9
- [9] Lee J, Adachi S, Peper F and Morita K 2003 Asynchronous game of life, in preparation
- [10] Hauck S 1995 Asynchronous design methodologies: an overview *Proc. IEEE* 83 69–93
- [11] Lee J, Peper F, Adachi S and Morita K 2002 Compact designs of universal delay-insensitive circuits with bi-directional and buffering lines in preparation
- [12] Martin A J 1990 Programming in VLSI: from communicating processes to delay-insensitive circuits *Developments in Concurrency and Communication (UT Year of Programming Institute on Concurrent Programming)* ed C A R Hoare (Reading, MA: Addison-Wesley) pp 1–64
- [13] C. P. Collier, E. W. Wong, M. Belohradský, F. M. Raymo, J. F. Stoddart, P. J. Kuekes, R. S. Williams, and J. R. Heath, "Electronically configurable molecular-based logic gates," *Science*, vol. 285, pp. 391–394, July 1999.
- [14] J. R. Heath, P. J. Kuekes, G. S. Snider, and R. S. Williams, "A defect-tolerant computer architecture: Opportunities for nanotechnology," *Science*, vol. 280, pp. 1716–1721, June 1998.
- [15] J. Chen, W. Wang, M. A. Reed, A. M. Rawlett, D. W. Price, and J. M. Tour, "Room-temperature negative differential resistance in nanoscale molecular junctions," *Appl. Phys. Lett.*, vol. 77, pp. 1224–1226, Aug. 2000.
- [16] J. Chen, M. A. Reed, A. M. Rawlett, and J. M. Tour, "Large on-off ratios and negative differential resistance in a molecular electronic device," *Science*, vol. 286, pp. 1550–1552, Nov. 1999.
- [17] Von Neumann J 1966 *Theory of Self-Reproducing Automata* ed AW Burks (Champaign, IL: University of Illinois Press)
- [18] Codd E F 1968 *Cellular Automata* (New York: Academic)
- [19] Serizawa T 1987 Three-state Neumann neighbour cellular automata capable of constructing self-reproducing machines - *Syst. Comput. Japan* 18 33–40
- [20] Reggia J A, Armentrout S L, Chou H-H and Peng Y 1993 "Simple systems that exhibit self-directed replication" - *Science* 259 1282–7