Generic Parallel Processing Techniques for Nanoscale Spin-Wave Architectures[1]

Mary M. Eshaghian-Wilner[2]
Electronics Practice, Foley & Lardner LLP
and Electrical Engineering Dept., UCLA

Shiva Navab
Broadcom Corporation
and Electrical Engineering Dept., UCLA

Abstract - *In this paper, we study the algorithm design aspects of three newly developed spin-wave architectures. The architectures are capable of simultaneously transmitting multiple signals using different frequencies, and allow for concurrent read/write operations. Using such features, we introduce a set of generic parallel processing techniques that can be used for routing and design of fast algorithms on these spin-wave architectures. We also present a set of application examples to illustrate the operation of the proposed generic parallel techniques.*

Keywords: parallel algorithms, parallel architectures, nanoscale spin-wave architectures.

1 Introduction

In mapping parallel algorithms onto parallel architectures, their performance highly depends on the underlying interconnectivity. In the past few decades, various architectures with different types of interconnectivity were introduced with the aim of producing fast algorithms. One measure that shows how fast algorithms can run on an architecture is the "diameter" of that architecture. The "diameter" of an architecture is defined as the minimum number of steps required to traverse the distance between the farthest pair of points on that given architecture. A diameter dictates a lower bound on the time complexity of running any non-trivial algorithm on a given architecture. For instance, the diameter of an N×N mesh is O(N), while it is O(log N) on an N×N mesh of trees, and O(1) on bus-based architectures [1].

Because of the low constant time diameter of bus-based architectures, many such architectures were designed in the 1980s and 1990s that aimed to produce very fast algorithms. However, the major shortcoming in most such bus-based designs has been that the use of buses was restricted to one processor at a time. In this paper, we deal with a set of newly developed architectures with spin-wave buses that allow multiple pairs of processors to intercommunicate over each bus at any given time.

Previously, researchers had studied various alternatives for replacing electrical buses. For example replacement of electrical interconnects with fiber optics was studied in [2,3], where they designed parallel architectures with fiber optical buses that used Time Division Multiplexing (TDM) and/or Wavelength Division Multiplexing (WDM). While electro-optical designs have their own challenges [4,5], these architectures alleviated some exclusive access restrictions of the bus and they were all implementable at micro-scale. The architectures presented here are nanoscale, yet are not subject to the exclusive access requirement on the buses.

In recent years, many nanoscale devices and computing methods have been developed [6-33]. The nanoarchitectures studied in this paper are spin-based and transmit waves as opposed to transmitting charge. These architectures allow a reduction in power consumption and provide a high level of interconnectivity between many more paths than was previously possible.

The idea of using spin-waves for both data transmission and data processing was initially introduced in [34]. Based on that, three parallel spin-wave architectures were developed that use spin waves for intercommunication, as well as for computation. The first architecture, presented in [35], is a spin-wave-based crossbar that interconnects multiple inputs to multiple outputs with spin-wave buses. The next architecture, presented in [36], is a spin-wave reconfigurable mesh. These two architectures require the same number of switches and buses as standard crossbars and reconfigurable meshes, but are capable of simultaneously transmitting multiple waves on each of the spin-wave buses, using dynamic frequency (or wavelength) division multiplexing. As a result, very fast and fault-tolerant algorithms can be designed for these architectures. The third architecture, presented in [37], is a fully interconnected cluster, consisting of functional units that communicate using spin waves. In this architecture, each

[1] Authors are listed in alphabetical order
[2] Mary M. Eshaghian-Wilner is a Patent Agent at the Electronics Practice Group of Foley & Lardner, LLP. She is also an Adjunct Professor of Electrical Engineering at the University of California, Los Angeles.

node can simultaneously broadcast to all other nodes, and can concurrently receive and process multiple data. This design allows all nodes to intercommunicate in constant time without increasing their fan-in and fan-out.

In this paper, we study the algorithm design aspect of these three newly developed spin-wave architectures. Exploiting their highly parallel features, we introduce a set of generic parallel processing techniques that can be used for the design of fast algorithms on these spin-wave architectures. We also present a set of application examples to illustrate the operation of the proposed generic parallel techniques on these new spin-wave models.

The rest of the paper is organized as follows: In Section 2, we present a brief overview of the three spin-wave-based architectures. In Section 3 we present generic operations for algorithm design on these architectures, followed by concluding remarks in Section 4.

2 Spin-Wave Architectures

In this section we give an overview of three spin-wave architectures: a spin-wave crossbar, a spin-wave reconfigurable mesh, and a spin-wave fully interconnected cluster. For more detailed explanations on these architectures please refer to [35, 36, 37] respectively.

2.1 Spin-Wave Crossbars

Crossbars are attractive architectures because they can realize any permutations of N inputs to N outputs. However, their main shortcoming is due to the fact that N^2 switches are used to transmit only N pairs of data. The architecture described here, while requiring the same number of switches as standard crossbars, is capable of transmitting N^2 data elements. This is because each spin-wave bus is capable of carrying multiple waves at any given time by using different frequencies. Therefore, each of the N inputs in parallel can essentially broadcast its data to all of the N outputs. As compared to molecular nano-scale crossbars, this design is more fault-tolerant because if there is a failure in one of the N channels, other channels can be used to transmit the data. This is possible because all the channels are accessible by all the processing nodes and each channel can handle multiple data.

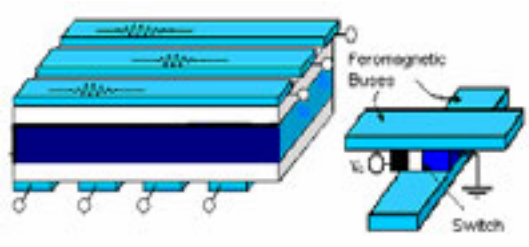

Figure 1 - Spin-Wave Crossbar Architecture

An example of the proposed spin-wave cross-bar architecture is shown in Figure 1. Note that a set of column spin-wave buses on the bottom and a set of row spin-wave buses on the top are connected via the vertical spin wave switches. A spin-wave switch is a device that has an externally controllable magnetic phase. In the "On" state, the switch transmits spin waves, while in the "Off" state it reflects any incoming spin wave.

As described in [35], the ferromagnetic film is divided by a region of diluted magnetic semiconductor (DMS), and it is used as a magnetic channel. The magnetic phase is controlled by the applied electric field via the effect of hole-mediated ferromagnetism. A negative gate bias increases the hole concentration in the DMS region, resulting in the paramagnetic-to-ferromagnetic or Off-to-On transition, whereas a positive bias has the opposite effect.

2.2 Spin-Wave Reconfigurable Mesh

A nanoscale reconfigurable mesh of size N^2 consists of an N× N array of processors connected to a reconfigurable spin-wave bus grid, where each processor has a locally controllable bus switch. An example of the proposed spin-wave reconfigurable mesh architecture is shown in Figure 2. Note that a set of column spin-wave buses on the bottom and a set of row spin-wave buses on the top are connected via the spin-wave switches.

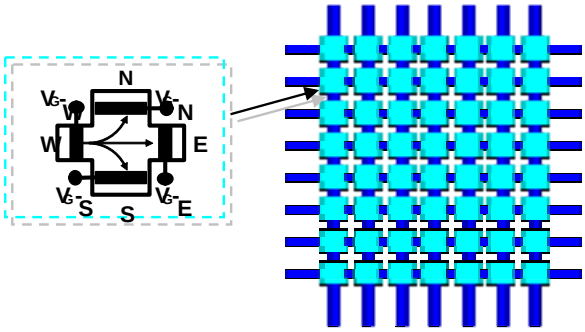

Figure 2 - The Nanoscale Reconfigurable Mesh

Each switch is placed at the grid point of the mesh. The switches allow the broadcast bus to be divided into sub-buses providing smaller reconfigurable meshes. These switches are similar to crossbar switches, except each reconfigurable mesh switch has four controllable gates to route the signal in different directions.

Basically, except for the spin-wave buses, the nanoscale reconfigurable mesh with spin-wave buses is similar to the standard reconfigurable mesh. It is worth noting that, similar to the reconfigurable mesh (and standard mesh), the nanoscale reconfigurable mesh of size N occupies N× N area, under the assumption that processors, switches, and a link between adjacent switches occupy unit area. However,

the main difference in terms of area here is that the unit of area is at nanoscale level as opposed to the standard reconfigurable meshes that are currently available at micro-scale level of integration.

2.3 Spin-Wave Fully Interconnected Cluster

A fully interconnected architecture consists of N computing nodes, all of which intercommunicate with spin waves. Figure 3 shows the top view of the architecture in which the N computing nodes are placed around a circle on a magnetic film. The area requirement of this architecture is $O(N^2)$ as opposed to the $O(N^4)$ area requirement if electrical interconnects were to be used. We should also note that all the distances in this architecture are at the nanoscale level.

Unlike electrical interconnection networks, in which only one transmission can be done at a time, here multiple simultaneous permutations are possible by transmitting the spin waves over different frequencies. The information is coded into the phase of the spin waves in the sender and is detected by the receivers. In addition, within each frequency, data can be sent to one or more other nodes from each node.

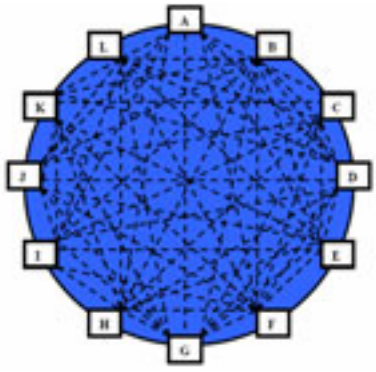

Figure 3 – Top View: Spin-Wave Fully Interconnected Cluster

Normally, in architectures where the phases of the waves are the means of information transmission, the exact location of the nodes with respect to the size of topology is an important design issue. The distance between the sender and receiver has to be at a length that is a multiple of the wave's wavelength; otherwise, the receiver might receive the wave with a π radian phase-shift, which is a "0" instead of a "1" or vice versa. However, in our design, this is not an issue because the wavelengths of spin waves are considerably larger than the distance between the nodes. The speed of spin waves is around 10^5m/s. Assuming the input frequency range of 1-10 GHz (as in our experiment), the wavelength will be in the order of 10^{-4} to 10^{-5}m, while the distances are nanoscale or 10^{-9}m. In other words, the wavelengths of the spin waves are some orders of magnitude greater than the distances between the nodes. Therefore, all the nodes receive the same phase regardless of their location, and there is no need to place the nodes in specific distance relative to the other ones.

3 Algorithm Design Techniques

In this section, we present three generic parallel processing techniques that are mainly designed for spin-wave reconfigurable meshes. The implementation of some of these techniques using spin-wave crossbar and fully interconnected clusters are also briefly discussed. When using these architectures, one should take into consideration the fact that two different types of data detections are possible at the nodes. Once the spin waves are detected by the receiver, the transmitted data can either be digitized or they can be left analog.

In analog detection mode, receiver detects a voltage that is the *superposition* of multiple waves. For example, if ten waves are sending a "1," then their analog sum through their cumulative amplitude is computed instantly as 10. Also, this property can be used to compute logical functions as described previously. In digital detection mode, this value is digitized to just a "1", and then the computations are continued digitally.

3.1 Finding the Minimum/Maximum

In this section, we first explain an implementation of an algorithm for finding the minimum and maximum of numbers on spin-wave architectures, and then present an image-processing algorithm as an example of an application that uses the minimum/maximum finding algorithm.

3.1.1 Implementation

To find the maximum or minimum of $O(N^2)$ inputs, with a maximum value of N, on a reconfigurable mesh, we first assign each value to a processing node on the grid points of the reconfigurable mesh. Next, each node checks its most significant bit (MSB). If MSB is 1, it broadcasts s'1' to all the other processing nodes. If MSB is 0, on the other hand, the node listens to the channel. If there is a 1 on the bus, that node gets disabled. At the next step, the nodes check their next most significant bit. This procedure is repeated for all the bits. Since the maximum value of the inputs is N, this procedure takes O(log N) time. If the number of inputs is less than or equal to N, the minimum and maximum can be found in O(1) as presented in [38].

Finding the minimum is similar to finding the maximum except that in the case of MSB=1, the node broadcasts a '1' and at the same time listens to the channel. If there is a 1 on the bus, that node becomes disabled.

The same implementation technique applies when using one single column of a spin-wave crossbar. In that case, the max/min of at most N inputs can be mapped to N processing units, and one of the columns would be used as the shared medium, where nodes broadcast their MSBs. All the switches on that column must be turned on to connect the processing units to the shared bus.

This algorithm can also be implemented on a fully interconnected cluster in a similar fashion, except that in the fully interconnected cluster there are no switches to be set. All the nodes are connected to the shared medium, where they can broadcast their MSB or receive signals from other nodes.

Note that on the three architectures, the whole routine can be performed simultaneously on disjoint sets of input, using different frequencies for each set. This cannot be done on a standard VLSI architectures due to conflicts on the buses.

3.1.2 An Application Example

Finding the maximum/minimum routine is used in designing several applications. As an example of image processing applications that use this routine, we present finding the nearest neighboring figure algorithm. The input to this algorithms is a digitized (black and white) image with processor (i,j) storing the pixel (i,j), $0 \leq i,j \leq N-1$, in the plane, where the black pixels are 1-valued and white pixels are 0-valued. We show that give an $N \times N$ image, using an $N \times N$ reconfigurable mesh with spin-wave buses, the nearest neighbor of a single figure can be found in O(log N) time.

Before finding the nearest neighbor, the figures need to be identified and labeled, and their convex hull should be found. The time complexity of the labeling algorithm is O(log N), as we presented in [39]. Convex hull of a figure can be found in O(log N) as well, as shown in the next section.

In order to find the nearest neighbor of a figure on a spin-wave reconfigurable mesh, we first set up the switches so that all nodes in the whole image are connected to each other. For all the extreme points of this figure, we should find the closest "1" belonging to another figure. For this, each extreme point, a node with at least one "0"-valued neighbor, broadcasts its address, as well as its ID, to all four directions. As soon as a black node from another figure (with a different ID) receives this broadcasted signal, it computes its distance with that node and sends it back to the original node. It also blocks that signal and does not let it pass though.

Using the technique explained above, we check the top, bottom, right, and left neighbors, but note that we may still miss some closer neighbors that are located diagonally from that figure. To address this issue, we need to revise the algorithm in the following way to use the white nodes as well:

All the black nodes (including non-extreme points) broadcast their address and ID to their neighbors, and whenever a black pixel receives the data, it blocks it. So now all the black (and white) nodes, know their distance from the closest neighbor from a different figure.

When an extreme point broadcasts its address, to the nearest neighboring black nodes in the four directions, all the intermediate white nodes that have received this broadcasted signal, find the distance between that node and their nearest black node having a different ID. On each side of the extreme point, the minimum distance of all these nodes is found in a binary tree fashion, e.g., each two adjacent nodes find their minimum that will be compared to the minimum of their adjacent pair, and so on. To find these minimum values, the max/min routine explained in the previous section can be used. The time complexity of this routine is O(log N). Therefore, all the extreme points will know their closest neighbor in O(log N) time. The next step will be to find the minimum number among the data of all the extreme points, which takes O(1) time. As a result, the overall time complexity of the nearest neighbor algorithm is O(log N).

3.2 Finding the First/Last in the List

In many applications, we need to find the first or last element of a list. In geometric algorithms, for instance, this operation is used to find the extreme points of a figure. A labeling algorithm presented in [39] is an application example that uses this routine several times. Here, we explain the routine and then present a new image processing application, namely finding the convex hull of a figure that uses this routine.

3.2.1 Implementation

To find the first or last element of at most N inputs, the first step is to store the inputs in the processing elements in a column. We find the local topmost nodes on that column by making each node send a signal downward. This procedure begins by turning off all the switches above each node (disconnect the channel) to force the signal sent by each node downward. Then all nodes simultaneously send out a signal downward and turn on all the switches (reconnecting the channel), thereby letting all signals pass through the bus. Each node that receives a signal will become disabled. As a result, the only active nodes are the local topmost nodes that have not received a signal. This procedure is performed in O(1) time. An example is shown in Figure 4, where the first "A" in the list is found by disabling the rest of the "A"s.

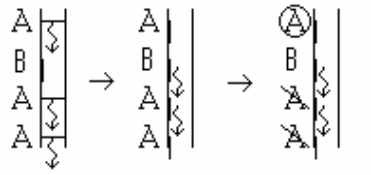

Figure 4 - Finding the First Element of a List

If different sets exist among the inputs, it is possible to find the first element of each set using frequency division multiplexing. For instance, in the above example, there are two sets (As and Bs). As shown in Figure 5, we find the first A and B in the list simultaneously by performing the routine on two different frequencies for A and B. This method is used in the graph formation algorithm for the partially ordered multiple-sequence alignment problem, presented in [40].

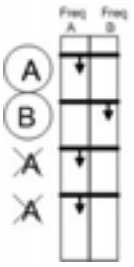

Figure 5- Finding the First Element of Different Sets

Unlike a reconfigurable switch that can be controlled in four different directions, a crossbar switch has just two possible statuses: "on" or "off". Therefore, it is not possible to force the wave in only one direction. Consequently, the first/last routine cannot be implemented on a crossbar in a similar fashion as a reconfigurable mesh. This implementation does not apply to a fully interconnected network either, because in that symmetric architecture there is no notion of first or last.

3.2.2 An Application Example

As an application example that uses the first/last element of the list routine, we explain the following convex hull algorithm. We show that give an N × N image, using an N× N reconfigurable mesh with spin-wave buses, the convex hull of a single figure can be found in O(log N) time.

This algorithm operates in two steps. First the rightmost (RM) and the leftmost (LM) black pixels ("1"-valued) in each row are found. Then it will be verified if these points are extreme points or not.

The "first in the list" routine is used to find the LM and RM black pixels or "1"-valued on each row. In other words, first all the left and right switches of each node are tuned on, so that all the channels between the nodes in the same row are open. Then each black node sends out a "1" signal on the bus and turns off its lefthand side switch, thereby closing that section of the channel and not letting the waves pass though from right to left. Closing that particular section belonging to the node will still allow the node to receive signals, but it will not let signals travel beyond (to the left) of that node. If at least one black pixel exists on a row, all the nodes on that row will receive a "1" except the LM node. Now that LM is identified in each row, we do the same to find the RM node. The only difference is to turn the righthand side switch and not let the waves go from left to right. Since setting all the switches in all the rows is done in parallel, finding RM and LM takes O(1) time.

The verification is done by having the ith column compute the enclosing angle made by the 1's in the image with the RM of the ith row (RM_i). This angle is defined to be the smallest angle Φi such that all the 1's are enclosed inside the region X RM_i Y as shown in Figure 6. For this, first all the RMs broadcast their address to all the 1's in their row. So now all the columns have a copy of all RM nodes. Each RM_i sends its address to all the black nodes in the ith column, and all these nodes compute their angle with RM_i. We need to find the maximum value (angle) of the nodes above RM_i and below it. RM_i is an extreme point if and only if the sum of these two angles is less than 180 degrees. This step is repeated for LM_i. The maximum angle value is found in O(log N) time using the max/min routine explained in the previous section. Therefore, the total time complexity of finding the convex hull of a figure will be O(log N). The O(1) time variation of this algorithm for a smaller image size has been presented in [38].

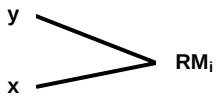

Figure 6 - Enclosing Angle X RMi Y

3.3 Finding the All Prefix Sum

The problem of all prefix is to compute the sum of first j elements in an array and store the sum in the jth element. Using this algorithm, the sum of all the input values will be computed and stored in the last element. In this section we show how this operation is performed in O(1) time on spin-wave architectures.

3.3.1 Implementation

To find the All Prefix Sum of at most N inputs, first we map them to the processing nodes on one column of the spin-wave reconfigurable mesh. At the beginning, all the switches are turned on. Then, each node sends a signal with an amplitude equal to the value it has stored, and

immediately turns off the switch above it to direct its generated wave downward. Note that for synchronization purposes, each node generates its spin-wave signal after a short period of time to compensate the spin-wave propagation delay in the ferromagnetic channel. This short period of time has to be equal or greater than the distance between adjacent nodes divided by the speed of spin waves, which is in the order of $10^{-9}/10^{4} = 10^{-13}$. Considering the fact that the frequency is in the order of GHz or even THz, this whole process is considered as constant time. Due to superposition property of spin-waves, the sum of previous values is found at each element. The main idea behind superposition is that when nodes transmit signals on the same frequency, the signals will superpose in amplitude when they meet. The amplitude of the superposed wave received by each node will be the prefix sum on that node. This routine is used in the graph formation algorithm for partial-order multiple-sequence alignment, presented in [40].

A special case of Prefix Sum routine is counting the number of elements that come before a certain element in the list. In this case, all the nodes transmit a signal downward with an amplitude of 1. In each node, the total amplitude of all the superposed signals sent by the nodes above shows the number of elements before that element in the list.

Another special case of this routine, would be the summation routine, where the summation of all nodes can be found in constant time. This is basically the Prefix sum routine at the last node of the list. Note that the Prefix sum routine can be performed on at most N inputs, whereas the summation can have as many as N^2 inputs, on all the nodes of the reconfigurable mesh. All the processing elements send a signal with an amplitude equal to the value they have stored. Utilizing the superposition property of spin waves, the summation of the input values is found. If the nodes tune their receiving frequency on their sending frequency, they will all receive the superposition or "sum" of the input values.

As described for first/last routine, it is not possible to send signals downward in a crossbar or in a fully interconnected cluster, so this routine can not be implemented on these two architectures. However, the special case of finding the summation of all inputs, for at most N inputs, can be implemented on one single column of a crossbar and on a fully interconnected cluster, as explained in max/min routine.

3.3.2 An Application Example

Consider the problem of creating a histogram of input values, each value representing a type, where all such values are in the range of 1..N. Assuming that the number of inputs is at most N^2, each input can be stored in one processing node on the spin-wave reconfigurable mesh. We show that this problem can be solved in O(1) time.

There are two approaches to storing the result of the histogram. First is to choose one node of each type as the representatives of that type. We can find the first element of each type in the list to represent its type, using the routine explained in the previous section. The second approach, which is the one we take, is to keep track of the number of each type in all the nodes of that type.

The next step is to count the number of times each value has been repeated in the list and store the sum at each (or the representative) node. As explained above, this problem is a subset of the "finding the sum" routine. In this case, all the nodes send a value of "1" and not an arbitrary value.

Note that this routine should be performed on each type of input; however, using different frequencies, these separate routines can be done simultaneously. In other words, each type's sender and receiver are tuned to a distinct frequency. Due to the superposition characteristic of the waves, each node receives the sum of "1"s sent on its tuned frequency, which is the number of nodes of that type.

4 Conclusion

In this paper, we studied the algorithm design aspect of three newly developed spin-wave architectures. The architectures are capable of simultaneously transmitting multiple signals using different frequencies, and allow for concurrent read/write operations. Exploiting such parallel features, we introduced a set of generic parallel processing techniques. We also presented a set of application examples to illustrate the incorporation of the proposed generic parallel techniques on these new spin-wave models.

5 References

[1] R. Miller, Q.F. Stout, "Portable parallel algorithms for geometric problems," Proceedings of 2nd Symposium on the Frontiers of Massively Parallel Computation, Page(s):195 - 198, Oct. 1988.

[2] J. W. Goodman, F. J. Leonberger, S. Y. Kung and R. A. Athale, "Optical Interconnections for VLSI Systems,"Proceedings of the IEEE, Vol. 72, No. 7, 1984.

[3] E. Harder, S.K. Lee, H. A. Choi, "On Wavelength Assignment in WDM Optical Networks," Proceedings MPPOI '97, Canada, June 1997

[4] H. S. Hinton, "Architectural Considerations for Photonic Switching Networks," IEEE Journal on Selected Areas in Communications, August 1988.

[5] M. M. Eshaghian, and L. Hai, "A Glance at VLSI Optical Interconnects: From the Abstract Modelings of the 1980s to Today's MEMS Implementations," book chapter "Handbook on Innovative Computing," 2006.

[6] M. M. Eshaghian-Wilner, A. H. Flood, A. Khitun, J. Fraser. Stoddart, and K. L. Wang., "Molecular and Nano-scale Computing and Technology," edited volume by Albert Zomaya, entitled "Handbook of Innovative Computing," Springer-Verlag (USA), 2006.

[7] P. Balasingam and V.P. Roychowdhury, "Nanoelectronic functional devices," Purdue University Technical Report: TR-EE 94-24, 1994.

[8] D. Goldhaber-Gordon, M.S. Montermerlo, J.C. Love, G.J. Opiteck, and J.C. Ellenbogen, "Overview of nanoelectronic devices," Proceedings of IEEE, 1997.

[9] J. Jortner and M. Ratner, Molecular Electronics, Oxford, U.K., (1997).

[10] Y. Luo, C.P. Collier, J.O. Jeppesen, K.A. Nielsen, E. DeIonno, G. Ho, J. Perkins, H.-R. Tseng, T. Yamamoto, J.F. Stoddart, and J.R. Heath, "Two-dimensional molecular electronics circuits," ChemPhysChem 3, 519, 2002.

[11] J.R. Heath and M.A. Ratner, "Molecular electronics," Physics Today, May, 43 (2003).

[12] Dmitri B Strukov and Konstantin K Likharev,"CMOL FPGA: a reconfigurable architecture for hybrid digital circuits with two-terminal nanodevices", Nanotechnology 16, p. 888–900, 2005.

[13] Lewin, D.I., "DNA computing," Computing in Science & Engineering, Volume 4, Issue 3, Page(s):5 – 8, 2002.

[14] J.H. Reif, "Parallel Biomolecular Computation: Models and Simulations," Algorithmica, special issue on Computational Biology, Vol. 25, No. 2, 142-176, 1999.

[15] James, D. K.; Tour, J. M. "Organic Synthesis and Device Testing for Molecular Electronics," Aldrichimica Acta, 39, 47-56, 2006.

[16] J. Fortes, M. Davis, J. Harris, and R. Figueiredo, "Nanoelectronic Adaptive Systems,"GOMACTech-2003, March 31 - April 3, 2003.

[17] Y. Ono and Y. Takahashi, "Single-electron pass-transistor logic: operation of its elemental circuit," IEDM Technical Digest 297 2000.

[18] A.Nojeh, A. Ural, F. Pease, and H. Dai," Electric-field-directed growth of carbon nanotubes in two dimensions," J. Vac. Sci. Technol. B 22, 3421-3425, 2004

[19] K. Yano, T. Ishii, T. Sano, T. Mine, F. Muri, T. Hashimoto, T. Kobayashi, T. Kure, and K Seki, "Single-electron memory for giga-to-tera bit storage," Proceedings of IEEE 87, 633, 1999.

[20] D.H. Kim, S.-K. Sung, J.S. Sim, K.R. Kim, J.D. Lee, B.-G. Park, B.H. Choi, S.W. Hwang, and D. Ahn, "Single-electron transistor based on a silicon-on-insulator quantum wire fabricated by a side-wall patterning method," Appl. Phys. Lett. 79, 3812 2001.

[21] A.C. Seabaugh and R. Lake, "Beyond-the-roadmap technology: Silicon heterojunctions, optoelectronics, and quantum devices," Encyclopedia Of Physics 22, 335 (1998).

[22] J.P. Sun, G.I. Haddad, P. Mazumder, and J.N. Shulman, "Resonant tunneling diodes: models and properties," Proceedings of IEEE 86, 641 ,1998.

[23] S.J. Tans, R.M. Verschueren, and C Dekker, "Room temperature transistor based on a single carbon nanotube," Nature 393, 49,1998.

[24] F. Leonard and J. Tersoff, "Negative differential resistance in nanotube devices," Physical Review Letters 85, 4767, 2000.

[25] P. G. Collins, M. S. Arnold, P. Avouris, "Engineering Carbon Nanotubes and Nanotube Circuits Using Electrical Breakdown", vol. 292, Science magazine, April 2001

[26] P. Avouris, J. Appenzeller, R. Martel, S. J. Wind, "Carbon Nanotube Electronics," Proceedings of the IEEE, vol. 91, NO. 11, Nov. 2003

[27] J. Phillips, K. Kamath, K.T. Brock, P. Bhattacharya, "Room temperature self-organized quantum dot transistors," 56th Annual Device Research Conference Digest, 1998. Page(s):46 - 47, June 1998.

[28] S.L. Chuang, N. Holonyak, "Quantum-well assisted tunneling injection quantum-dot lasers," Lasers and Electro-Optics Technical Digest, Page(s):297 vol.1,2002.

[29] G. Snider, A. Orlov, I. Amlani, X. Zuo, G. B. Stein, C. Lent, J. Mez, and W. Porod, "Quantum-dot cellular automata," Journal of Applied Physics 1999.

[30] C. S. Lent, S. E. Frost, P. M. Kogge, "Reversible computation with quantum-dot cellular automata (QCA), Computing Frontiers Conference, 2005.

[31] D.D. Awschalom, M.E. Flatté and N. Samarth, "Spintronics," Scientific American May (2002).

[32] A. Khitun, R. Ostroumov, and K.L. Wang. Spin-wave utilization in a quantum computer, Physical Review A, 64(6): p. 062304/1-5, 2001.

[33] A. Khitun, R. Ostroumov, and K.L. Wang, "Feasibility study of the spin wave quantum network," 10th International Symposium on Nanostructures: Physics and Technology, Proceedings of the SPIE, volume 5023, pp. 449-451, 2003.

[34] A. Khitun, and K. L. Wang, "Nano Scale Computational Architectures with Spin-wave Bus, Superlattices & Microstructures. 38(3), 184-200, 2005.

[35] M. M. Eshaghian-Wilner, A. Khitun, S. Navab, and K. L. Wang, "A Nanoscale Crossbar with Spin Waves," Proceedings of the 6th IEEE Conference on Nanotechnology, Ohio, USA, July, 2006.

[36] M. M. Eshaghian-Wilner., A. Khitun, S. Navab, and K.L. Wang, "A Nano-Scale Reconfigurable Mesh with Spin Waves," The ACM International Conference on Computing Frontiers, Italy, May 2006.

[37] M. M. Eshaghian-Wilner., A. Khitun, S. Navab, and K.L. Wang, "Nano-Scale Modules with Spin-Wave Inter-communications for Integrated Circuits," The NSTI Nanotech 2006, Boston, MA, May 2006.

[38] M. M. Eshaghian-Wilner, A. Khitun, S. Navab, and K. L. Wang, "A Nano-scale Architecture for Constant Time Image Processing," to appear in a special issue of the "physica status solidi" (a) Journal, 2006-2007

[39] M. M. Eshaghian-Wilner, A. Khitun, S. Navab, and K. L. Wang, "The Spin-wave Nano-Scale Reconfigurable Mesh and the Labeling Problem," ACM Transaction on Emerging Technology, 2006-2007.

[40] M. M. Eshaghian-Wilner, L. Lau, S. Navab, and D. Shen, "Parallel Graph Formations of Partial-Order Multiple-Sequence Alignments Using Nano-, Micro-, and Multi-Scale Reconfigurable Meshes", submitted for publication.