SUPERPOSED PARALLEL BUSES:

A SYSTOLIC AREA-TIME OPTIMAL VLSI INTERCONNECTION

Bruce W. Arden

Toshio Nakatani

CS-TR-019-86

February, 1986

Superposed Parallel Buses: A Systolic Area-Time Optimal VLSI Interconnection

Bruce W. Arden and Toshio Nakatani

Department of Computer Science
Princeton University
Princeton, N.J. 08544

ABSTRACT

This paper presents a new class of interconnection schemes, based *on superposed parallel buses* and called *systolic grid interconnection.* This scheme is optimal both in area and time in terms of VLSI complexity for multi-point networks with prescheduled permutation. This scheme is faster than previously reported interconnection schemes using linear area. This scheme is especially suitable for utilizing fast but area-limited technology to interconnect a large number of single chip processors using slower technology. Although the primary proposed application is to solve large sparse systems of linear equations, such as those arising from finite element analysis, this interconnection scheme is widely applicable because of its fast permutation capability. The construction of the grid and its VLSI optimality are described as well as some applications.

Feb 05, 1986

Superposed Parallel Buses: A Systolic Area-Time Optimal VLSI Interconnection

Bruce W. Arden and Toshio Nakatani

Department of Computer Science
Princeton University
Princeton, N.J. 08544

1. Introduction

The finite element formulation of physical problems can typically be represented as interacting computations on the nodes of a planar graph, and generally such problems are solved using iterative methods such as the one due to Jacobi (Adams[1982]). This method requires $\Omega(N)$ area since the bisection width (Thompson[1979]) of a family of the finite element graphs with N nodes is $O(\sqrt{N})$ (Lipton and Tarjan[1979]).* Leiserson[1983] has shown that $O(N\log^2 N)$ area is sufficient for layout, but he observes that he knows of no planar graph that requires more than linear area. However, optimal mapping of a finite element graph to a fixed grid is intractable, as is shown by Bokhari[1981]. The difficulty of direct mapping has been described in several different contexts. The Finite Element Machine (FEM) developed by NASA (Jordan[1978]) has a hexagonal mesh of processors augmented by a global bus, and it has suffered from the long latency of global communication. The modification (Dew[1984]) of two dimensional Systolic Matrix Multiplication Arrays (Kung[1979]) for Jacobi iteration by the insertion of delay queues is a second example. It works nicely for banded matricies, but not for non-banded matricies. It also requires frequent dynamic reconfigurations of the delay queues, which would interrupt the data flow and degrade the system performance.

An alternative is to use a class of rearrangeable, multistage interconnection networks to customize the processor interconnections for the topology of the

---

* 1) There exist an infinite number of planar graphs that require $O(NlogN)$ area (Ullman[1984a]).

2) In this paper, "$f(n)=O(g(n))$" means that there is some constant c so that $f(n)\leq cg(n)$ for all sufficiently large n. Similarly, "$f(n)=\Omega(g(n))$" means that there is some constant c so that $f(n)\geq cg(n)$ for all sufficiently large n. "$f(n)=\Theta(g(n))$" means that $f(n)=O(g(n))$ and $f(n)=\Omega(g(n))$.

given finite element graph. This is based on the idea that the connectivity of a sparse graph with maximum degree d can be simulated by d permutation routings, which can be easily shown using Hall's[1935] distinct representative theorem. However, as Rosenberg[1983] has shown, it requires $\Theta(N^2)$ area to implement such networks because of many crossing wires between stages. The NYU Ultracomputer (Schwartz[1980]), or its successor the IBM RP3 (Pfister[1985]), uses an Ω-network, which is a blocking network that is one half of the Benes rearrangeable network (Lawrie[1975]). The interconnection for 512 processors expends a large area to lay out the shuffle wires. The IBM GF11 Supercomputer (Beeten[1984]), which is intended for QCD computations and uses three-stage Benes rearrangeable networks to interconnect 576 processors, must also accommodate the large number of wires for shuffle connections. All the parallel processors interconnected by the shuffle or hypercube connections, whether they use a packet switching or a circuit switching scheme, share the same disadvantage; there are area penalties in planar layouts. This could become a major limitation to the interconnction of large numbers of processors. In particular, the quadratic growth of wire area would prohibit the use of fast technology for the processor interconnections, and yet fast technology would be indispensible for compensating the $\Omega(logN)$ theoretical lower bound in the graph diameter to interconnect N processors. This is obvious because fast technology limits the chip area more severely than the slower technology. This is obvious, for example, if we consider the use of ECL and GaS technology today. It seems that this relation between feature size and speed will continue. Furthermore, pin limitations greatly restrict the design of large interconnection networks. It is possible to reduce the shuffle connections by using larger crossbars as the basic building blocks. For example, the GF11 uses 24×24 crossbars. But, it will be much harder to design a large VLSI crossbar (Makrucki[1981], Rau[1981], Georgiou[1984], Hsu[1984]). If we assume fully parallel (as oposed to serial) operation, an $n\times n$ crossbar requires at least $\Omega(n)$ data pins and $\Omega(n\log n)$ control pins (Thurber[1971]). The requirement of this large number of pins limits the use of large crossbars.

On the other hand, microprocessors are inherently local and well suited to chip implementation. RISC processors (Patterson[1981], Hennessy[1981], Radin[1983]) are popular examples where the processing elements and associated registers fit nicely on a chip. This observation serves to emphasize the need for a compact interconnection scheme to exploit the computational chips for parallel

computation. It is obvious from their function that interconnection components are inherently non-local and pose the problems described above. The solution lies in maximizing the traffic on a relatively small numbers of links - much less than that required for complete, simultaneous single-step communication. A promising approach, therefore, is to simulate the pin-limited crossbars by highspeed time-multiplexed buses (we call its clock cycle *a minor cycle*) and to simulate the area-consuming shuffle connections by rotation of data movement, while preserving the powerful permutation capability of rearrangeable multistage networks and maintaining high communication bandwidth for the processors (we call a processor's cycle *a major cycle*). Careful selection of the minor cycle to the major cycle ratio would clearly increase the speedup of parallel algorithms.

In this paper, we present *a systolic grid interconnection* for statically scheduled permutations. It is optimal both in area and time in terms of VLSI complexity for multipoint networks (Ullman[1984b]), and therefore suitable for employing fast but area-limited technology. The network can permute data from N processors in $\sqrt{N}$ minor cycles in a systolic fashion, and $3\sqrt{N}$ minor cycles without systolic synchrony. This is faster than any other interconnection scheme using $O(N)$ area known to us. As an example of the interconnection schemes requiring linear area, the mesh-connection scheme requires at least $4(\sqrt{N}-1)$ steps as a lower bound for permutation and sorting, as Thompson and Kung[1977] have shown. Thompson[1978] has given a $(7\sqrt{N}-8)$ time routing method for prescheduled permutations by applying a binary Benes rearrangeable network to a mesh-connected processor. For sorting, Kung and Thompson[1977] (also Nassimi and Sahni[1979]) have given a $6\sqrt{N}$ algorithm on a mesh-connected processor by applying a Batcher's[1968] bitonic sort. Unlike the others, the permutations are not prescheduled since it is a "real time" sorting algorithm. By comparison, the grid interconnection has the fastest permutation time using linear area. Furthermore, if a highspeed time-multiplexed bus is used to send at least one packet of data from each of $\sqrt{N}$ processors to a destination processor on the same bus within $\sqrt{N}$ minor cycles, and if one major (processor) cycle is equivalent to $\sqrt{N}$ minor cycles, then N processors can permute their data in every major (processor) cycle. In the following sections, we describe the construction of such an interconnecton, a systolic architecture, routing algorithms, universality, VLSI optimality and some application areas.

2. Construction of Grid Interconnections for Fast Permutation

To construct a fast permutation network, we transform a three-stage $N=n^2$ Benes-Clos (Benes[1962], Clos[1953]) rearrangeable network (Figure 2.1) into a two dimensional $n \times n$ square grid of *superposed parallel buses.* First, we replace each $n \times n$ crossbar by a highspeed time-multiplexed linear bus, which can transmit at least one packet of data from each processor (a total of n processors on a bus) to the other on the same bus in $n=\sqrt{N}$ minor cycles. Second, we overlay the second column of n buses onto the first column of n buses by a rotation of 90 degrees, which eliminates the shuffle connection between the first and the second columns of crossbars. Similarly, we overlay the third column of n buses onto the second column by the same rotation. Finally, we get the two dimensional square grid of *superposed parallel buses* (Figure 2.2). Permutations can be performed in three steps by simulating the original Benes rearrangeable network. That is, at each step, a crossbar function can be simulated by the time-multiplexd linear buses in the sense that the broadcast address and data are received by the listeners in the prescheduled order of the listener's address. A shuffle function, between steps, can be simulated by switching the broadcasting buses from the row to the column, and from the column to the row.* A total of three steps are required to simulate the whole three-stage rearrangeable network. The performance of the interconnection can be further improved by pipelining the three stages. If one major (processor) cycle is equivalent to $\sqrt{N}$ minor cycles, then every processor can send to a destination and receive from a source a packet on every processor cycle. Furthermore, this scheme can be extended to a systolic architecture as described in the following sections. In summary, we obtain the following theorem:

Theorem 2.1: With a $n \times n$ grid interconnetion, an arbitrary prescheduled permutation can be completed every $3n$ minor cycles. If the operation is systolic, then an arbitrary prescheduled permutation can be completed every n minor cycles.

Proof: The grid interconnection simulates the three-stage Benes permutation network in $3n$ minor cycles.□

Interconnection schemes requiring linear area, such as mesh-connected processors where every processor has only local connections with four neighbors,

* A similar observation has been made in the context of three dimensional embeddings of a Banyan network (Wise[1981]) and a permutation network (Preparata[1983]).

have been studied extensively since the ILLIAC-IV (Barnes[1968], Orcutt[1974]). The lower bound for sorting and permutation on a $N=n^2$ mesh-connected processor is $4(n-1)$ steps, as Thompson and Kung[1977] have shown. They (and Nassimi and Sahni[1977]) have also given a $6n$ time algorithm for sorting. Furthermore, Thompson[1978] has shown a $(7n-8)$ time routing method for prescheduled permutation. Nassimi and Sahni[1980] have shown an optimal routing algorithm for frequently used permutations on a mesh-connected processor, but not for arbitrary permutations. Recently, Stout[1983] and Kumar[1985] have shown that some operations of a mesh-connected processor can be improved by augmenting the row and colum broadcasting buses. However, they have not suceeded in improving permutation with the augmented buses. The grid interconnection proposed here appears to be the fastest prescheduled permutation scheme implemented in linear area.

3. A Systolic Architecture

In order to pipeline the three stages of permutation operations described in the previous section and to implement a square grid of the highspeed time-multiplexed buses using area limited technology, a systolic broadcasting chip (Figure 3.2) has been outlined. A set of four chips is used to link a processor to three sets of buses; a total of $4n^2$ chips (32 bits for data and 12 bits for tags) are required to construct the entire $n \times n$ *systolic grid interconnection* (Figure 3.3) for permutation operations of n^2 processors. At $3n$ minor cycles after every processor starts sending a packet, entire registers on the systolic broadcasting chips will be filled with the packets and n row buses $R_{0,0}$–$R_{0,(n-1)}$, n column buses $C_{0,0}$–$C_{0,(n-1)}$, and n row buses $R_{1,0}$–$R_{1,(n-1)}$ will operate systolically (Figure 3.4 shows the scheduling chart). A packet (Figure 3.1) is composed of the *tag*0, *tag*1, and *tag*2 followed by the *data*. The broadcasting sequence is predetermined by the order of the contents of the tags and the program in each chip. Whenever the content of the leading tag of a packet is matched by the modulo n clock counter, a chip broadcasts its packet from the output register after stripping the leading tag. Whenever the modulo n counter is matched by its own turn that has been programmed in advance, the chip moves a packet from the bus into the input register.*

* The program, here, is a simple identification number representing the turn for the chip to take a broadcast data into its input register, and is independent of the specific permutations.

For example, a packet with *tag*=0 will be broadcast first to the row bus and a packet with *tag*=1 will be broadcast second to the row bus, etc. The content of *tag*0 determines the order of broadcasting a packet to the row buses $R_{0,0}$–$R_{0,(n-1)}$, *tag*1 the order to the column buses $C_{0,0}$–$C_{0,(n-1)}$, and *tag*2 the order to the row buses $R_{1,0}$–$R_{1,(n-1)}$. Every processor sends its packet to the first systolic broadcasting chip and receives its incoming packet from the fourth systolic broadcasting chip. Every time a packet is broadcast, the leading tag is consumed by a systolic broadcasting chip. That is, at first, *tag*0.*tag*1.*tag*2.*data* is sent from a processor and received by the first chip. Second, *tag*1.*tag*2.*data* is broadcast by the first chip and received by the second chip. Third, *tag*2.*data* is broadcast by the second chip and received by the third chip. Fourth, *data* is broadcast by the third chip and received by the fourth chip. Finally, the designated data is ready at the fourth chip to the destination processor. These operations proceed systolically to perform the prescheduled permutation among n^2 processors. If a highspeed bus can broadcast n packets within one major (processor) cycle, then all the n^2 processors can send and receive the prescheduled permutation of data every processor cycle.

4. Routing Algorithms

Routing algorithms for rearrangeable multistage networks have been extensivly studied (Waksman[1968], Opferman and Tsao-Wu[1971], Andresen[1977], Lee[1981]). These algorithms all run in $O(N\log N)$ time on a single processor. The natural extension of these algorithms to a parallel system leads to a $O(N)$ time parallel algorithm (Parker[1978]). The faster parallel algorithms have been developed by Nassimi and Sahni[1982] and Lev, Pippenger, and Valiant[1981]. These parallel algorithms run on N completely connected processors in $O(log^2 N)$ time.* The completely connected processors can be simulated by sorting data on the single stage shuffle-exchange network (Stone[1971]) in $O(log^2 N)$ time slowdown. Therefore, routing tags for the permutations of the faster parallel algorithms can be calculated on N processors with a systolic grid interconnection in $O(log^4 N)$ time. The routing for specific permutations, which might be called *a "boot-strap"*, can be done with the shuffle-exchange (Stone[1971]) as the default interconnection.

* Here we assume that $n=\sqrt{N}$ is an integral power of two. Otherwise, time complexity of these algorithms for a three-stage $n \times n$ Benes-Clos network becomes $O(log^3 N)$.

5. Universality

Since the original Benes rearrangeable network can perform an arbitrary permutation by setting the switches with a precalculated control pattern, the $n \times n$ systolic grid interconection can also simulate any n^2 single-stage fixed interconnection network with fan-in/out degree d in dn minor cycles. If a highspeed bus for a grid interconnection can transmit n packets to n processors on a bus in one major cycle, then a $n \times n$ systolic grid interconnection can simulate any n^2 single-stage fixed interconnection network with fan-in/out degree d in d major cycles. As a summary, we have the following theorem:

Theorem 5.1: With a $n \times n$ systolic grid interconnection, any n^2 single-stage fixed interconnection network with fan-in/out degree d can be simulated in dn minor cycles.

Proof: Any n^2 single-stage fixed interconnection network with fan-in/out d can be simulated sequentially by d permutation routings.□

6. VLSI Optimality for Permutation

Ullman[1984b] has extended Thompson's[1979] AT^2 lower bound argument for the point-to-point networks to the AT^2M^2 lower bound for the multipoint networks and has shown that a fast sorting network can be constructed using high "*flux*" nets. For sorting, he has shown the lower bound $AT^2M^2=\Omega(N^2)$ and the lower bound of the maximum number of processors on a net, $M=\Omega(\sqrt{N})$, which is necessary to achieve the optimal fast sorting networks using $O(N)$ minimum area. By these criteria, the so-called grid-of-nets has been shown to be nearly optimal as a fast sorting network. In similar fashion, we can show that our interconection scheme is optimal for permutation by choosing $M=\sqrt{N}$. For an N-permuter, Rosenberg[1983] has shown that $AT^2=\Omega(N^2)$ and $A=\Omega(N^2)$. We can extend this formula to the AT^2M^2 lower bound for N-permuter as follows:

Theorem 6.1: For an N-permuter,

(1) $AT^2M^2=\Omega(N^2)$ and $A=\Omega(N)$

(2) $AT^2M^2=O(N^2)$ and $A=O(N)$

Proof: (1) The information transfer required is $\Omega(N^2)$ for permutation. (2) Since the grid interconnection can permute N data items in $T=O(1)$ time and $A=O(N)$ area, the network can achieve the lower bound, provided the highspeed time-multiplexed bus is fast enough to support $M=\sqrt{N}$ processors on the bus.□

Therefore, the interconnection scheme is optimal as a permuter for both time and area in terms of VLSI complexity for the multipoint network.

7. Applications

A systolic grid interconnection is suitable for those synchronous parallel algorithms, whose communication patterns are known prior to execution and therefore the necessary permutations can be precalculated. Moreover, since no special memory is required to store the control patterns for permutations, as in the GF11 supercomputer (Beeten[1984]), the number of permutations is unlimited. Although finite element computations are particularly well matched by this architecture, arbitrary sparse matrix computations are also efficiently performed. Furthermore, any statically schedulable dataflow program with large parallelism can be effectively executed on this architecture by precalculating the permutations necessary for connectivity of granules at every synchronization point. The important architectural issue here is to determine the relative communication complexity and arithmetic complexity of the parallel algorithms under consideration. Observations have been made for some numerical parallel algorithms (Gannon and Rosendale[1984] and Johnsson[1985]). In the context of a parallel processor with a grid interconnection, correct selection of the ratio of the minor cycle to the major cycle is the important design factor for efficient execution of the chosen family of parallel algorithms. In addition, we further note that the size of computational granules obviously affects the relative effects of communication complexity and arithmetic complexity.

For example, a saturated (Gottlieb and Kruskal[1984]) parallel Jacobi algorithm for finite element graph with maximum degree d requires d permutations, while every iteration requires roughly d multiplication, (d–1) additions, and one division. For effective pipeline operation of computation and communication, the computation time of each iteration should be equal to the communication time for the next iteration. That is, if we assume that any arithmetic operation takes one major cycle, then the selection of technology for interconnection must be done in such a way that one major cycle is equal to n minor cycles for $n \times n$ systolic grid interconnection. This is because computation time for each iteration is roughly equal to the communication time, on condition that the multiply unit and addition unit are pipelined.

On the other hand, a supersaturated (Gottlieb and Kruskal[1984]) parallel Jacobi algorithm on the n^2 parallel processors requires $O(\frac{\sqrt{K}}{n})$ permutations for

a finite element graph with K nodes, since the bisection width of planar graphs with K nodes is $O(\sqrt{K})$, while a computation at every iteration comprises of $O(\frac{K}{n^2})$ arithmetic operations. In this case, a period of $O(\frac{n^2}{\sqrt{K}})$ minor cycles is close enough to one major cycle to achieve a linear speedup of the number of processors, n^2. That is, the bisection width of a sparse graph affects the communication complexity when the granule size increases and therefore affects the relative effects of arithmetic complexity and communication comlexity. A suitable selection of the effective ratio of the minor cycle to the major cycle for the computational granules under consideration is a key for a successful parallel computation with a systolic grid interconnection.

8. Conclusions

We presented an area-time optimal interconnection scheme, called *systolic grid interconnection,* for arbitrary prescheduled permutation. Unlike the rearrangeable multistage networks, this interconnection has neither pin limitations due to crossbars nor quadratic growth of shuffling area, and yet preserves fast permutation capability. Unlike the fixed interconnection network, it has no mapping problem from a parallel algorithm to the network. Furthermore, owing to its linear and regular area layout, we can utilize fast technology for the interconnection and thereby time-multiplex the highspeed broadcast operation with a very fast minor cycle on the bus. By selecting the optimal ratio of the minor cycle to the major cycle, efficient parallel computations can be carried out with a systolic grid interconnection.

We note that transformation of space-division networks to time-division networks has been discussed (Marcus[1972], Andresen and Harrison[1972]) in the context of space-time equivalence in communication networks for the cost of the number of cross points. We have further shown that a crossbar can be completely eliminated by a highspeed time-multiplexed bus and that a shuffle interconnection can be eliminated by a rotational transformation from row to column and from column to row. These observations lead to a linear area layout. The *systolic grid interconnection* scheme is particulary important when we consider VLSI implementaion of interconnections of a large number of processors. That is, the number of cross points is no longer a valid criterion for the cost of interconnection networks. VLSI layout area and pin limitations are more important factors for both the cost and the performance of interconnection networks. The

larger the number of processors connected, the more important the interconnection area becomes.

Future research in this area includes an extension of grid interconnections to dynamic permutation capability for real-time applications. A randomized, local routing strategy (Valiant[1981], Aleliunas[1982], Upfal[1982]) might be helpful for asynchronous communication. Another alternative is to find an oblivious routing strategy (Borodin and Hopcroft[1982]) that is not strictly local but communication-synchronized or when-determinate (Lipton and Sedgewick[1981] and Ullman[1984a]). For the latter strategy, we have investigated the simulations of an Ω-network on the systolic grid interconnection for dynamic permutation (Arden and Nakatani[1986]). Although it requires additional hardware to implement, the systolic grid interconnection can successfully simulate an Ω-network.

References

Adams, L. [1982]. "Iterative algorithms for large sparse systems on parallel computers," 166027, NASA Langley Research Center Report.

Aleliunas, R. [1982]. "Randomized Parallel Computation," *ACM Symposium on Principles of Distributed Computing* 1982, pp. 60-72.

Andresen, S. and S. R. Harrison [1972]. "Toward a general class of time-division multiplexed connection networks," *IEEE Trans. on Communications* COM-20:5, pp. 836-846.

Andresen, S. [1977]. "The looping algorithm extended to Base 2^t rearrangeable switching networks," *IEEE Trans. on Communications* COM-25:10, pp. 1057-1063.

Arden, B. W. and T. Nakatani [1986]. "Permutations on superposed parallel buses," Technical Report CS-24, Department of Computer Science, Princeton University.

Barnes, G.H. et al. [1968]. "The ILLIAC IV computer," *IEEE Trans. on Computers* C-17:8, pp. 746-757.

Batcher, K. E. [1968]. "Sorting networks and their applications," *1968 Spring Joint Comput. Conf., AFIPS Conf. Proc.,* 32, Washington, DC, pp. 307-314.

Beeten, J., M. Denneau, and D. Weingarten [1984]. "The GF11 Supercomputer," IBM Computer Science Research Report RC-10852.

Benes, V. E. [1962]. "On rearrangeable three-stage connection networks," *Bell Sys. Tech. J.* 41, pp. 1481-1492.

Bokhari, S. H. [1981]. "On the mapping problem," *IEEE Trans. on Computers* C-30:3, pp. 207-214.

Borodin, A. and J. E. Hopcroft [1982], "Routing, merging, and sorting on parallel models of computation," *Proc. Fourteen Annual ACM Symposium on the Theory of Computing,* pp. 338-344.

Clos, C. [1953]. "A study of non-blocking switching networks," *Bell Sys. Tech. J.* 32, pp.406-424.

Dew, P. M. [1984]. "VLSI architeture for problems in numerical computation," *Supercomputers and Parallel Computation* Oxford University Press, New York, pp. 1-24.

Gannon D. B. and J. V. Rosendale [1984]. "On the impact of communication complexity on the design of numerical algorithms," *IEEE Trans. on Computers*

C-33:12, pp. 1180-1194.

Georgiou, C. J. [1984]. "Fault-tolerant crosspoint switching networks," IBM Computer Science Research Report RC-10446.

Gottlieb, A. and C. P. Kruskal [1984]. "Complexity results for permuting data and other computations on parallel processors," *J. ACM* 31:2, pp. 193-209.

Hall, P. [1953]. "On representatives of Subsets, *J. London Math. Soc.,* 10, pp. 26-30.

Hennessy, J., J. Norman, F. Baskett, and J. Gill [1981]. "MIPS: a VLSI processor architecture," In Kung, H. T., R. Sproull, and G. Steele(eds.), *VLSI Systems and Computations,* Computer Science Press, Rockville, Md.

Hsu, F.H., T. H. Kung, T Nishizawa, and A. Sussman [1984]. "LINC: The link and interconnection chip," Technical Report, Carnegie-Mellon University, Computer Science Dep., May, 1984.

Johnsson, S. L. [1985]. "Data permutation and basic linera algebra computations on ensemble architectures," Research Report YALEU/DCS/RR-367, Yale University.

Jordan, H. F. [1979]. "A Multiprocessor system for finite element structural analysis," *Computers and Structures,* 10, pp 21-29.

Kung, H. T. [1979]. "Let's design algorithms for VLSI systems," *Proc. of Caltech Conf. on VLSI: Architecture, Design,Fabrication,* pp. 65-90.

Kumar, V. K., and C. S. Raghavendra [1985]. "Array processor with multiple broadcasting," *Proc. of the twelvth Annual Symposium on Computer Architecture,* pp. 2-10.

Lawrie, D. H. [1975]. "Access and alignment of data in array processor," *IEEE Trans. on Computers,* C-24:12, pp. 1145-1155.

Lee, K. Y. [1981]. "On the rearrangeability of a 2 logN-1 stage permutation network," *Proc. of the Inter. Conf. on Parallel Processing* Silver Spring, Md., IEEE Computer Society Press, pp. 221-228.

Leiserson, C. E. [1983]. *Area efficient VLSI computation,* MIT Press, Cambridge, Mass.

Lev, G. F., N. Pippenger, and L. Valiant [1981]. "A fast algorithm for routing in permutation networks," *IEEE Trans. on Computers* C-30:2, pp. 93-100.

Lipton, R. J. and R. E. Tarjan [1979]. "A planar separator theorem," *SIAM J. Applied Math.* 36:2, pp.177-189.

Lipton, R. J. and R. Sedgewick [1981]. "Lower bound for VLSI," *Proc. Thirteenth Annual ACM Symposium on the Theory of Computing,*

Makrucki, B. A. and T. N. Mudge [1981]. "VLSI design of a crossbar switch," SEL Report No. 149, Dep. of Electrical and Computer Engineering, University of Michigan.

Marcus, M. [1972]. "Space time equivalents in connecting networks," *Conf. Record, 1972 IEEE Int. Conf. Communications* 31, pp. 35/25-31.

Nassimi, D. and S. Sahni [1979]. "Bitonic sort on a mesh-connected parallel computer," *IEEE Trans. on Computers* C-27:1, pp. 2-7.

Nassimi, D. and S. Sahni [1980]. "An optimal routing algorithm for mesh-connected parallel computers," *J. ACM* 27:1, pp. 6-29.

Nassimi, D. and S. Sahni [1982]. "Parallel algorithms to set up the Benes permutation network," *IEEE Trans. on Computers* C-31:2, pp. 148-154.

Opferman, D. C. and Tsao-Wu [1971]. "On a class of rearrangeable switching networks," *Bell Sys. Tech. J.* 50, pp. 1579-1618.

Orcutt, S. E. [1976]. "Implementation of permutation functions in illiac IV-type computers," *IEEE Teans. on Computers* C-25:9, pp. 929-936.

Parker, D. S. [1978]. "Huffman tree construction, nonliner recurrences, and permutation networks," Ph.D. dissertation, Dep. Computer Science, Univ. of Illinois, Urbana.

Patterson, D. A. and C. H. Sequin [1981]. "A reduced instruction set VLSI computer," *Proc. of The Eighth Annual Symposium on Computer Architecture* Minneapolis, Minn.

Pfister, G. F. et al. [1985]. "The IBM reaserch parallel processor (RP3) introduction and architecture," IBM Computer Science Research Report RC-11060.

Preparata F. P. [1983]. "Optimal three-dimensional VLSI layouts," *Math. Systems Theory* 16, pp. 1-8.

Radin, G. [1983]. "The 801 minicomputer," *IBM Journal of Research and Development,* 27:3, pp. 237-246.

Rau, B.R., P. J. Kuekes, and C. D. Glaeser [1981]. "Statically scheduled VLSI interconnect for parallel processors," In Kung, H. T., R. F. Sproull, and G. L. Steel(editors), *VLSI Systems and Computations,* pp. 389-395. Computer Science Press, Rockville, Md.

Rosenberg, A. L. [1983]. "Three-dimensional VLSI: a case study," *J. ACM* 30:3, pp. 397-416.

Schwartz, J. T. [1980]. "Ultracomputers," *ACM Trans. on Programming languagees and Systems* 2:4, pp.484-521.

Stone, H. S. [1971]. "Parallel processing with the perfect shuffle," *IEEE Trans. on Computers* C-20:2, pp.153-161.

Stout, Q. F. [1983]. "Mesh-connected computers with broadcasting," *IEEE Trans. on Computers* C-32:9, pp. 826-830.

Thompson, C.D., H. T. Kung [1977]. "Sorting on a Mesh-connected Parallel Computer," *Comm. ACM* 20:4, pp. 263-271.

Thompson, C.D. [1978]. "Generalized connection networks for parallel processor interconnection," *IEEE Trans. on Computers* C-27:12, pp. 1119-1125.

Thompson, C.D. [1979]. "A complexity theory for VLSI," Ph.D. dissertation, Carnegie-Mellon Univ., Pittsburgh, PA.

Thurber, K. J. [1971]. "Permutation switching networks," *Proc. 1971 Computer Designer's Conference,* Anaheim, Ca, pp. 7-24.

Ullman, J. D. [1984a]. *Computational Aspects of VLSI,* Computer Science Press, Rockville, Maryland.

Ullman, J. D. [1984b]. "VLSI complexity and supercomputers," *Proc. Fourth Jerusalem Conf. on Information Technology,* May, pp.646-649.

Upfal, E. [1982]. "Efficient scheme for parallel communication," *Proc. ACM Symposium on Principles of Distributed Computing* 1982, pp. 55-59

Valiant, L.G. and G. J. Brebner [1981]. "Universal schemes for parallel communication," *Proc. Thirteenth Annual ACM Symposium on the Theory of Computing,* pp. 263-277.

Waksman, A. [1968]. "A permutation network," *J. ACM* 15, pp. 159-163.

Wise, D. S. [1981]. "Compact layouts of Banyan/FFT networks," *Proc. of the CMU Conference on VLSI Systems and Computations,* Pittsburgh, Penn., pp. 186-195.

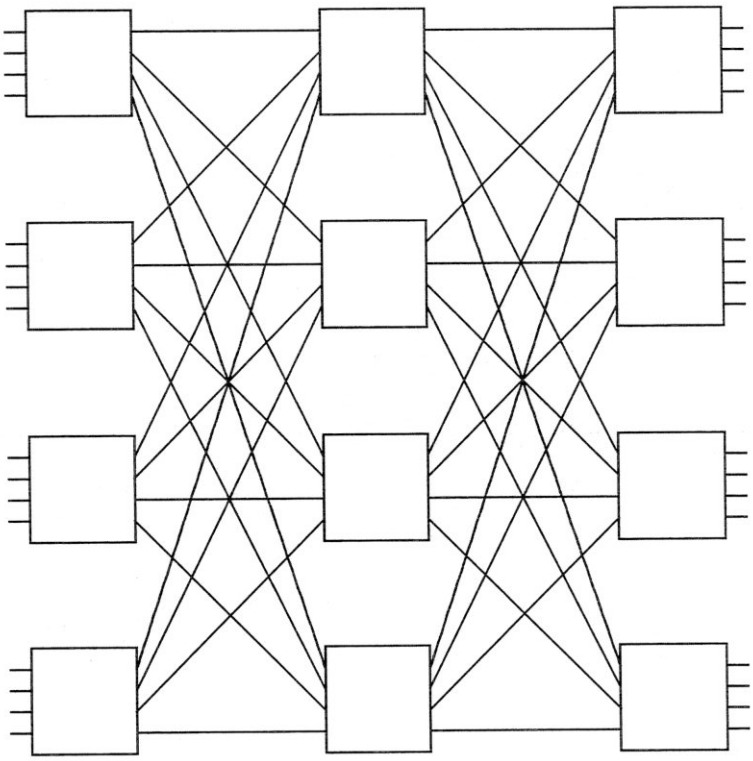

Figure 2.1: A three-stage 4x4 Benes-Clos rearrangeable network

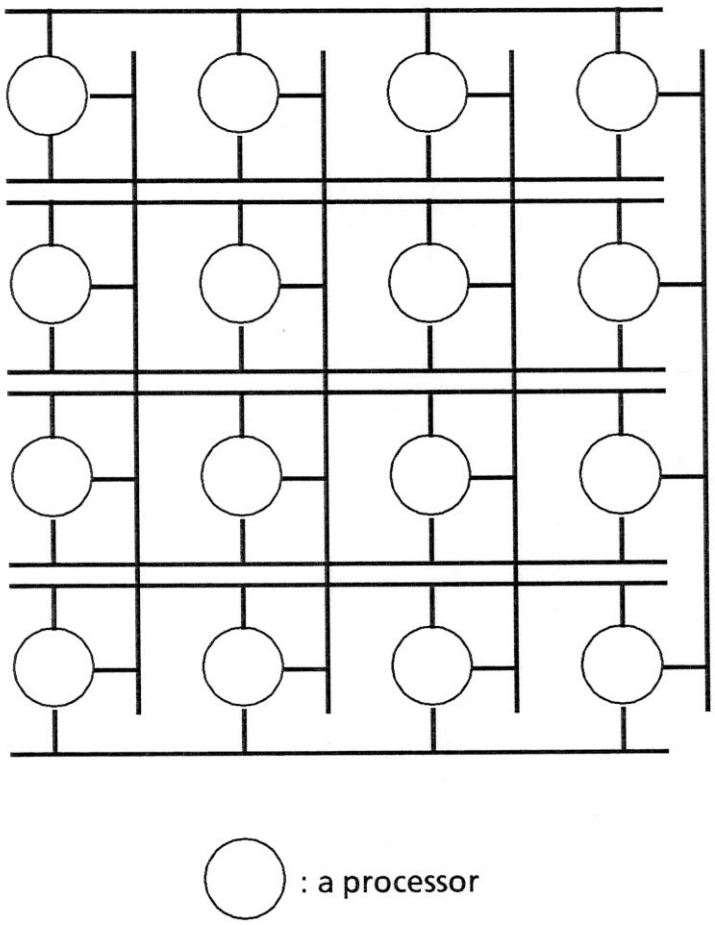

Figure 2.2: A 4x4 square grid of superposed parallel buses for permutations

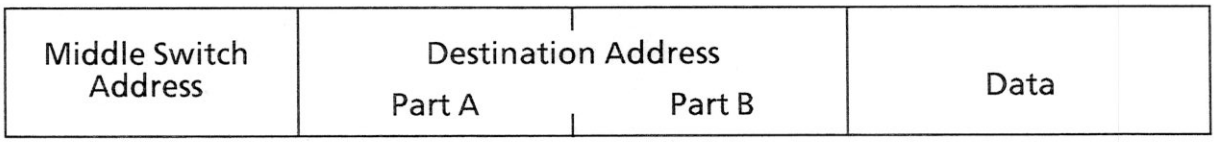

Figure 3.1: A packet format for permutations

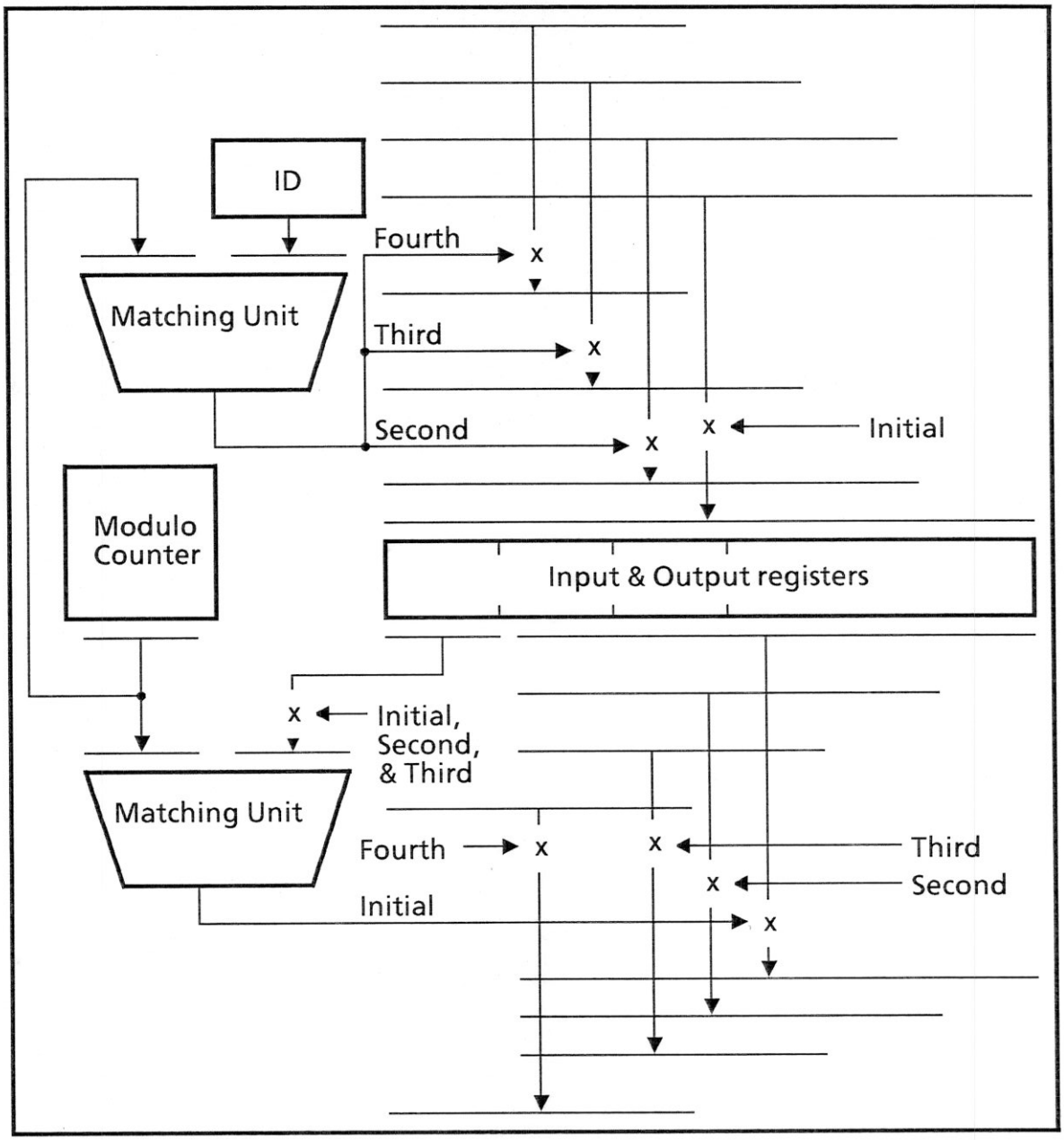

Figure 3.2: A broadcasting chip for permutations

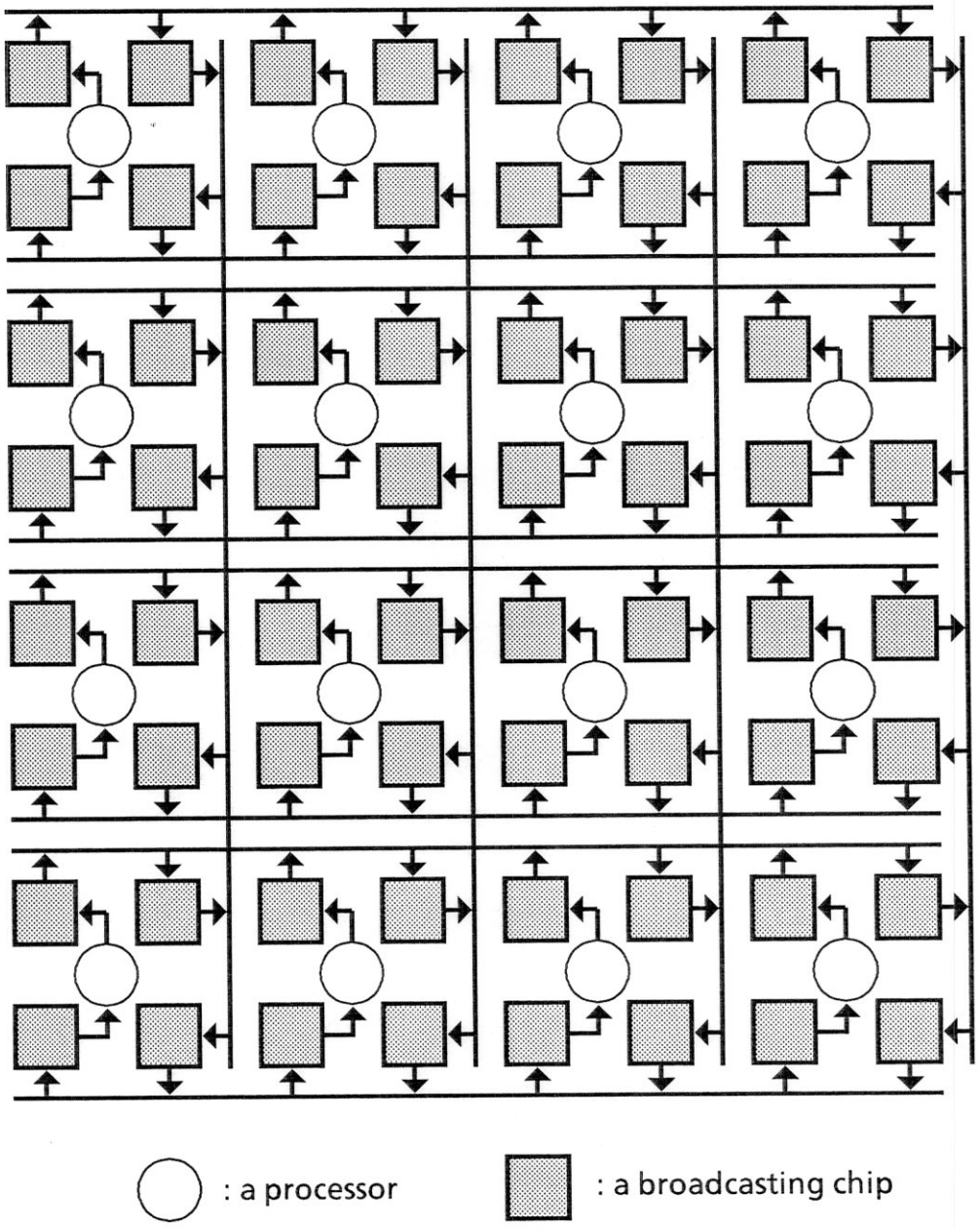

Figure 3.3: A systolic grid interconnection for permutations

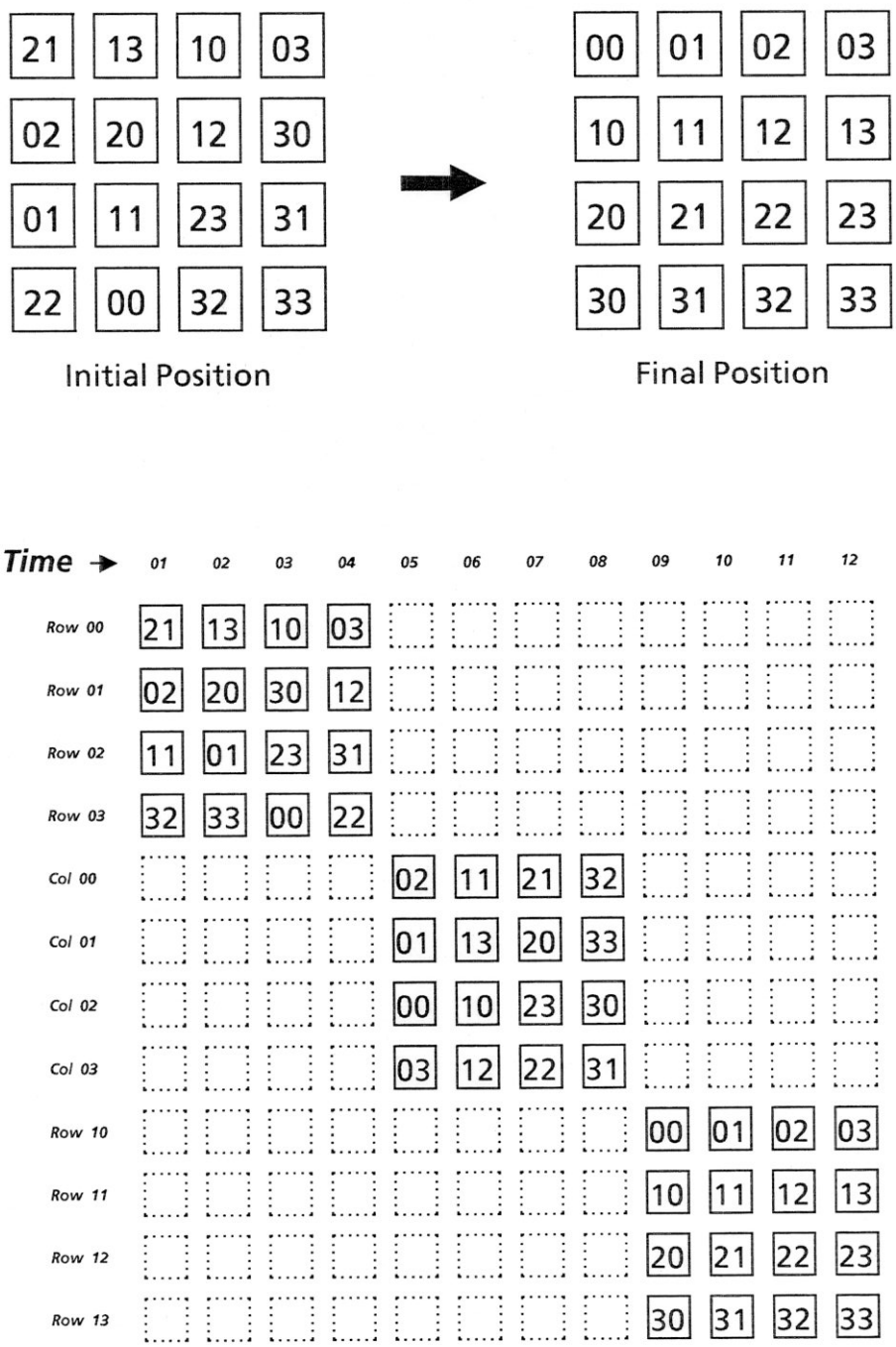

Figure 3.4: A scheduling chart for a 4x4 permutation