# A NEW APPROACH TO

# FAST CONTROL OF PERMUTATION NETWORKS

Bruce Arden

Abdou Youssef

CS-TR-032-86

April, 1986

# A New Approach to
# Fast Control of Permutation Networks

Bruce Arden

and

Abdou Youssef

## ABSTRACT

Controlling Benes-Clos networks in full generality has proven to be very slow. Many approaches have been taken to speed up the control for certain classes of permutations. In this report, a new approach is developed for 3-stage networks with nxn switches as building blocks (denoted by B(n,2)). The new approach allows the network to be self-routed for many interesting classes of problems, or more precisely, the permutations they need.

The families of permutations that can be self-routed by the new scheme are characterized. Several problems such as FFT, bitonic sorting, simulation of standard fixed interconnection networks on B(n,2), and others are shown to produce families of permutations that yield to the new scheme. A new control algorithm of $O(\log^2 N)$ complexity is derived for B(n,2) where $N = n^2$ is the number of terminals, and possible implementation of the scheme is discussed.

## I. INTRODUCTION:

Due to the advent of the VLSI technology, it has become feasible to build machines of hundreds and even thousands of processing elements (pe), each of which has a moderate computational power. These pe's are interconnected in some fashion to allow efficient communication between them. Depending upon whether these pe's are globally or locally controlled, these machines are called SIMD or MIMD machines [3]. In either class, performance is hinged on how efficiently the pe's can communicate with one another. Therefore, the fashion in which the processing elements are interconnected is of most concern.

It is possible to build optimal interconnection networks (IN) for classes of problems, that is, networks that provide a direct link between two pe's when they need to communicate. However, the only universal optimal IN's are the complete nets (i.e., each pe is connected to each other pe), or crossbar interconnections, which are prohibitively expensive. The need then arises for less costly, versatile and efficient networks, fixed or reconfigurable. Some of the standard fixed IN's are mesh networks, six neighbor networks, eight neighbor networks, hypercubes, cube connected cycles, trees, shuffle-exchange networks, etc. ... The reconfigurable networks are usually built with stages of crossbar switches, interconnected in a defined manner. Every time two terminals need to communicate, some of the switches are set in some way to establish the connection. Some of the standard reconfigurable networks are Benes-Clos networks [2], Batcher networks [1], omega networks [3]. It is worthwhile to note here that a good way to carry out the communication is by means of permutations. A permutation $\pi$ represents the fact that $pe_i$ wants to send data to $pe_{\pi(i)}$. To pass $\pi$ through the network, the switches

have to be set in such a way that each $pe_i$ is connected to $pe_{\pi(i)}$.

Both classes of networks, fixed or reconfigurable, need control algorithms, that is routing algorithms. These algorithms can incur additional overhead, and therefore, should be fast and efficient. There evolves a clear tradeoff between hardware and control time. To illustrate, Benes networks are near optimal in hardware but have slow control time, whereas Batcher networks use more hardware than Benes (less than crossbar nets, though) but can be controlled "on the fly". Omega networks, on the other hand, are cheaper than the other two nets and can be controlled on the fly, too; however, they do not have the same communication power as Benes or Batcher nets.

Thus, Benes nets emerge as the most hardware efficient reconfigurable IN's with full communication power, that is, they can pass all permutations. Yet, the known control algorithms for them are slow [5] and [8] and no self-routed algorithm exists. To remedy this, special efficient algorithms can be devised to pass subsets of interesting permutations that come up in the parallel algorithms studied in the literature. Several such algorithms have been found [4] and [6].

Three approaches to this problem can be identified. First, designing self-routed algorithms that use local data such as destination tags, stage number and switch number within stage to set each switch. Such algorithms can pass small subsets of permutaions, preferably interesting ones. This approach was taken by Nassimi ans Sahni [6i]. Second, identifying families of permutations that appear often in the literature and have wide application, then use any peculiar properties they have so that new, faster control algorithms can be found for them. This approach was taken by Lenfant [4]. The first approach produces optimal control but enables only a small fraction of the possible permutations to pass. The second approach offers more potential for larger families of permutations but still suffers speed inefficiency and implementation overhead. A third approach is needed, an

approach that combines the positive properties of the previous two approaches and avoids their limitations. It consists of fixing enough stages so that the remaining network can be self-routed and can pass the required permutations. This approach requires that, for each class of problems, the family of permutations be identified and checked to see if it can be passed with enough switching stages fixed so that the remaining stages can be set by a fast, self-routing algorithm. (such families are called compatible).

In this report, we will characterize the compatible families of permutations, and show that the families of permutations of FFT and bitonic sorting and other interesting problems are compatible.

The rest of the report is organized as follows: In the next section, some preliminaries and definitions are given for Benes and omega networks. Section III discusses the power of the scheme. In section IV, we characterize the compatible families of permutations (i.e., those that fit the scheme). Section V and VI show that the families of permutations of FFT and bitonic sorting are compatible. Section VII discusses the use of bitonic sorting to build interconnection networks that are as cheap in hardware as omega networks, with full communication power and fast control. In section VIII, standard fixed IN's are optimally simulated on Benes networks by compatible families of permutations. In section IX, we attempt to generalize, and give some open questions. Finally, we conclude with some remarks and possible implementation of the new scheme in section X.

## II. PRELIMINARIES AND DEFINITIONS:

In this section, we give some definitions related mostly to Benes-Clos and omega networks of different building blocks, and give also some preliminary facts about them.

Definition 1: An NxN Benes-Clos network, where $N = 2^k$, with 2x2 cross-bar switches as building blocks, is defined recursively as pictured in fig. 1.

Note that the indices before the first stage and after the last one represent the pe's. Note also that the number of stages is 2k-1 and that of switches is (2k-1)N/2.

Generalization 1: An NxN Benes-Clos network (denoted B(n,k) where $N = n^k$) with nxn crossbar switches as building blocks is also defined recursively as depicted in fig. 2.

The number of stages is 2k-1 and that of switches is (2k-1)N/n.

An important special case is when $N = n^2$. There are 3 stages , each of which has n nxn switches. The network when $N = 16 = 4^2$ is given in fig. 6.

Definition 2: An NxN omega network ($\Omega(2,k)$), where $N = 2^k$ and the building block is a 2x2 crossbar switch, consists of k consecutive stages, each of which is a shuffle connection and a column of N/2 switches, as shown in fig. 3 for the case $N = 2^3$.

Note that the shuffle (S) is a permutation of the set $R_N = \{0,1,...,N-1\}$ such that $S(i) = 2i$ if $0 \leq i \leq N/2 - 1$ and $S(i) = 2i - N + 1$ if $N/2 \leq i \leq N-1$.

Generalization 2: An NxN omega networks can be similarly generalized when $N = n^k$ to form $\Omega(k,n)$ of k stages, where the building block is an nxn switch and the shuffle S generalized in a natural way.

Definition 3: Omega inverse ($\Omega^{-1}$) is an $\Omega$ network except that the sources are conceived as destinations and vice versa.

In the remainder of this section, we will give some facts about these networks which will be needed later on. For a thorough study of omega networks, see [3], and for Benes-Clos nets, see [2]. We assume first that the stages are numbered 0, 1 , ... , k-1 (resp. 0, 1, ... , 2k-2) from left to right in omega networks (resp. Benes networks), and that the switches are numbered 0, 1, ... , N/n - 1 from top to bottom.

Fact 1: There is exactly one path between each source and each destination in omega nets.

Fact 2: Omega nets are self-routed as follows: Each source gives a destination tag (DT) (i.e., the index of the pe to receive that data). It is assumed that no two destination tags are identical. For $\Omega(2,k)$, the destination tag is a binary number $a_0a_1...a_{k-1}$. A switch in stage i examines $a_i$ and bi of the upper and lower DT, respectively. If $a_i = b_i$, there is a conflict and the permutation cannot be passed; otherwise, the switch is crossed or set straight through, depending on whether $a_i$ is 1 or 0, respectively.

In the general case ($\Omega(n,k)$ for $N = n^k$), the destination tag is still of the form $a_0a_1...a_{k-1}$ but $a_i$ is an n-ary digit (i.e., $0 \leq a_i \leq n-1$); a switch in stage i receiving n tags examines their i-th digits. If any 2 or more of them are identical, there is a conflict; otherwise, the switch is set as an nxn crossbar connection would be.

Fact 3: If the first k-1 stages of B(n,k) are set to identity (i.e., straight through), then the resulting network is equivalent to $\Omega(n,k)$.

Fact 4: If the last k-1 steges of B(n,k) are set to identity, then the resulting network is equivalent to $\Omega^{-1}(n,k)$.

Fact 5: If the first k-1 stages of B(n,k) are fixed to any configuration, the resulting network can be self-routed in the same way as described in fact 2. Note here that stage $k + i$ $(0 \leq i \leq k-1)$ is treated as stage i in fact 2. The resulting network is a "modified" omega. Precisely, it is $h\Omega$ for some permutation h.

Fact 6: In the important special case where $N = n^2$, fixing any of the three columns of B(n,2) enables the resulting network to be self-routed.

The main concern in this report is:

**Given a family of permutations, can we fix the first column of B(n,2) to some configuration in such a way that the resulting network can pass the whole family without conflict?**

If the answer is positive, we fix the first column and the network is self-routed thereafter for the duration of the whole family. Some new definitions and notations will be useful in dealing with these families of permutations. We assume that $N = n^2$ in the rest of the report unless stated otherwise.

Denote by $R_N$ the set $\{0, 1, \ldots, N-1\}$ of the indices of the pe's. Any number x in $R_N$ is uniquely written as $pn + q$ where $0 \leq p, q \leq n-1$.

Let H be the set of permutations (of $R_N$) that are admissible by any column of switches of B(n,2), by f the interconnection (permutation of $R_N$) between the first and second column of B(n,2). The interconnection between the second and third column is $f^{-1}$.

It is straightforward to prove that $f(pn + q) = qn + p$ by induction on p and q. This yields that $f = f^{-1}$.

Observe that H is a subgroup of the symmetric group $S_N$ (i.e., the group of

permutations of $R_N$), and h is in H if $h(pn + q) = pn + q'$ for some q' function of p and q.

The stages of B(n,2) are numbered 0, 1, 2 from left to right, and the switches in each stage are numbered 0,1, ... , n-1 from top to bottom.

If h is in H, we say that some stage of B(n,2) is set to h (or h-configured) if the p-th switch of that stage is set in such a way that the q-th source port of that switch is connected to its $h(pn + q)$-th destination port. Naturally, if a destination tag appears at input port $pn + q$ (i.e., the q-th input port of switch p), it comes out from the stage at output port $h(pn + q) = pn + q'$ (q'-th output port of switch p).

Definition 4: A permutation is said to be admissible by a network if the network can pass it without conflict.

Definition 5: A permutation is said to be h-admissible for some h in H if B(n,2) can pass it with column 0 set to h.

Definition 6: Let $\Phi_1, \Phi_2, \ldots, \Phi_m$ be m permutations in $S_N$. $\Phi_1, \Phi_2, \ldots, \Phi_m$ are compatible if there is some h in H such that $\Phi_1, \Phi_2, \ldots, \Phi_m$ are all h-admissible. Then $\Phi_1, \Phi_2, \ldots, \Phi_m$ are called h-compatible.

Definition 7: If t(p,q) is a mapping from $R_n x R_n$ to $R_n$, we define t(p,.) to be the mapping from $R_n$ to $R_n$ such that $t(p,.)(q) = t(p,q)$. We define t(.,q) similarly.

To conform to the topology of B(n,2), we put a parameter x on the left side of a composition of mappings of the sort $h_0 f h_1 f h_2$ where $h_0$, $h_1$, and $h_2$ are in H, but we reserve the liberty to put it on the left or right side of a simple mapping since no confusion arises.

Fact 7: If column 0 of B(n,2) is h-configured, then the resulting network is hΩ.

## III. USE AND POWER OF hΩ NETS:

As mentioned earlier, the advantage of hΩ is self-routedness, but a question arises about its power. The following shows the extent of hΩ's power.

$|h\Omega| = (n!)^{2n}$ (where |E| denotes the cardinality of the set E). To show this, note that $|h\Omega| = |\Omega|$ because we are working in a group, namely, $S_N$.

H is the set permutations passed by column 0. Each switch can pass n! permutations. Thus, n switches pass $(n!)^n$ permutations and the two stages of Ω pass $(n!)^{2n}$ permutations.

note that $|h\Omega|/|S_N| = (n!)^{2n}/(n^2)! \rightarrow 0$ as n goes to infinity. Thus, hΩ is a small fraction of $S_N$.Nevertheless, we can change h to our liking, adding versatility and more potentiality to include in our scheme many interesting classes of problems . It is in this freedom of selecting h where the power of our approach lies. As will be seen later, many known numerical and semi-numerical parallel algorithms discussed in the literature give rise to compatibe families of permutations, something that proves the power and benifit of our scheme.

## IV. CHARACTERIZATION OF COMPATIBLE PERMUTATIONS:

Each class of problems, such as sorting, FFT, matrix computations etc. ..., gives rise to a family of permutations needed to carry out the communication required if that class is run on a Benes-connected machine. Of most interest is to know whether a given family of permutations is h-compatible for some h in H, and if so, find such an h. The following is a theorem that characterizes compatible families. We start with a few lemmas, first.

Lemma 1: If $\Phi$ is a permutation, there exist $h_0$, $h_1$, $h_2$ in H such that $\Phi = h_0 f h_1 f h_2$.

Proof: B(n,2) passes all permutations in $S_N$ [2], and any configuration of any of the three columns of B(n,2) is in H.Therefore, $S_N$ = HfHfH. Q.E.D.

Lemma 2: If $\phi$ is in $S_N$, there exist two mappings $\alpha$ and $\beta$ from $R_n x R_n$ to $R_n$ such that $\phi(pn+q) = \alpha(p,q)n + \beta(p,q)$.

Proof : Define $\alpha(p,q) = \lfloor \phi(pn+q)/n \rfloor$ and $\beta(p,q) = \phi(pn+q) - \alpha(p,q)n$. It is easy to see that $\alpha$ and $\beta$ are two mappings from $R_n x R_n$ to $R_n$.

Lemma 3: h is in H iff there exists $t : R_n x R_n \rightarrow R_n$ such that $h(pn+q) = pn + t(p,q)$ and t(p,.) is a permutation of $R_n$ for every p,q in $R_n$.

Proof: Only if:

Define $t(p,q) = h(pn+q) - pn$. It is obvious to see that t is a mapping from $R_n x R_n$ to $R_n$, and t(p,.) is a permutation of $R_n$ for every p.

If:

It is enough to show that h is one-to-one.

If $h(pn+q) = h(p'n+q')$, then $pn + t(p,q) = p'n + t(p',q')$, which implies that $p = p'$ and $t(p,q) = t(p',q')$. Thus, $t(p,.)(q) = t(p,.)(q')$ and

therefore $q = q'$ because $t(p,.)$ is a permutation. Hence $pn + q = p'n + q'$. Q.E.D.

Note that if a column is set to h, then its p-th switch is set to $t(p,.)$.

Lemma 4: If $\phi$ is in $S_N$ and $\phi = h_0 f h_1 f h_2$ for some $h_0$, $h_1$, $h_2$ in H, and if $t_0$ is such that

$h_0(pn + q) = pn + t_0(p,q)$ and $\phi(pn + q) = \alpha(p,q)\, n + \beta(p,q)$, then

$h_1(t_0(p,q)n + p) = t_0(p,q)n + \alpha(p,q)$ and

$h_2(\alpha(p,q)n + t_0(p,q)) = \alpha(p,q)\, n + \beta(p,q)$.

Proof: $(pn + q)h_0 f h_1 f h_2 = (pn + t_0(p,q))f h_1 f h_2 = (t_0(p,q)n + p)h_1 f h_2 =$

$(t_0(p,q) + p')f h_2 = (p'n + t_0p,q))h_2 = p'n + q'$ for some $p'$ and $q'$ in $R_n$.

Since $\phi = h_0 f h_1 f h_2$, $p'n + q' = \alpha(p,q)\, n + \beta(p,q)$ and therefore $\alpha(p,q) = p'$ and $\beta(p,q) = q'$. Hence:

$h_1(t_0(p,q)n + p) = t_0(p,q)n + p' = t_0(p,q)n + \alpha(p,q)$ and

$h_2(\alpha(p,q)n + t_0(p,q)) = h_2(p'n + t_0(p,q)) = p'n + q' = \alpha(p,q)\, n + \beta(p,q)$.

Q.E.D.

Characterization theorem: Let $\phi_1, \phi_2, \ldots, \phi_r$ be r permutations in $S_N$. $\phi_1, \phi_2, \ldots, \phi_r$ are compatible iff there is $t : R_n x R_n \to R_n$ such that

(i) $t(p,.)$ is a permutation of $R_n$ for every p in $R_n$.

(ii) If for some i $\alpha_i(p,q) = \alpha_i(p',q')$ and $p \neq p'$, then $t(p,q) \neq t(p',q')$, where $\alpha_i(p,q) = \lfloor \phi_i(pn + q)/n \rfloor$.

Proof: The if part. We have $t : R_n x R_n \to R_n$ satisfying (i) and (ii).

Let $h_0$ be the following permutation of H: $h_0(pn + q) = pn + t(p,q)$.

After lemma 3, $h_0 \in H$.

Define $h_1^{(i)}$ and $h_2^{(i)}$ for i = 1, 2, ... , r as follows: $h_1^{(i)}(sn + p) = sn + \alpha_i(p,q)$ where $q = (t(p,.))^{-1}(s)$

and $h_2^{(i)}(\alpha_i(p,q)n + t(p,q)) = \alpha_i(p,q)n + \beta_i(p,q) = \phi_i(pn + q)$.

Since $t(p,.)$ is a permutation of $R_n$, then $(t(p,.))^{-1}$, the inverse mapping, exists and is a permutation too.Thus, $h_1^{(i)}$ is well defined as a mapping from $R_n$ to $R_n$.

$h_1^{(i)}$ is one-to-one:

$h_1^{(i)}(sn + p) = h_1^{(i)}(s'n + p')$

$\Rightarrow sn + \alpha_i(p,q) = s'n + \alpha_i(p',q')$ where $q = (t(p,.))^{-1}(s)$

and $q' = (t(p',.))^{-1}(s')$,

$\Rightarrow s = s'$ and $\alpha_i(p,q) = \alpha_i(p',q')$ after hypothesis (ii).

We claim that $p = p'$ because otherwise

$t(p,q) = t(p,.)(q) = s$ and $t(p',q') = t(p',.)(q') = s'$ and $s = s'$,

contradicting (ii). Therfore, $p = p'$, and consequently, $sn + p = s'n + p'$.

Hence, $h_1^{(i)}$ is one-to-one, a property that is enough to make $h_1^{(i)}$ a permutation of $R_N$ because $R_N$ is finite.

$h_2^{(i)}$ is a permutation:

We have to prove that $h_2^{(i)}$ is well-defined and bijective.

By well-defined we mean that if $\alpha_i(p,p)n + t(p,q) = \alpha_i(p',q')n + t(p',q')$ then $h_2^{(i)}((p,q)n + t(p,q)) = h_2^{(i)}((p',q')n + t(p',q'))$.

$\alpha_i(p,q)n + t(p,q) = \alpha_i(p',q')n + t(p',q') \Rightarrow \alpha_i(p,q) = \alpha_i(p',q')$

and $t(p,q) = t(p',q') \Rightarrow p = p'$ by hypothesis (ii).

$p = p'$ and $t(p,q) = t(p',q') \Rightarrow t(p,.)(q) = t(p,.)(q') \Rightarrow q = q'$ because $t(p,.)$ is one-to-one. Thus, we have $p = p'$ and $q = q'$.

$\Rightarrow \alpha_i(p,q)n + \beta_i(p,q) = \alpha_i(p',q')n + \beta_i(p',q')$

$\Rightarrow h_2^{(i)}(\alpha_i(p,q)n + t(p,q)) = h_2^{(i)}(\alpha_i(p',q')n + t(p',q'))$

Hence, $h_2^{(i)}$ is well-defined, and the same argument proves that $h_2^{(i)}$ is one-to-one where it is defined. So if we prove that $h_2^{(i)}(R_N) = R_N$, then $h_2^{(i)}$ should be totally defined because $|h_2^{(i)}(R_N)| = |\{x \in R_N | h_2^{(i)}(x)$ is defined$\}|$.

For every x in $R_N$, there exists $pn + q$ in $R_N$ such that $\phi_i(pn + q) = x$ because $\phi_i$ is a permutation.

$\phi_i(pn + q) = \alpha_i(p,q)n + \beta_i(p,q) \Rightarrow x = \alpha_i(p,q)n + \beta_i(p,q)$.

But $h_2^{(i)}((\alpha_i(p,q)n + t(p,q)) = \alpha_i(p,q)n + \beta_i(p,q) = x$.

Therefore, $h_2^{(i)}(R_N) = R_N$, and $h_2^{(i)}$ is well-defined, totally defined, and one-to-one $\Rightarrow h_2^{(i)} \in H$.

Now, it is a simple matter to prove that $\phi_i = h_0 f h_1 f h_2$ for i = 1, 2, ... , r.

Thus, $\phi_1, \phi_2, \ldots, \phi_r$ are $h_0$-compatible.

The only if part. We have $\phi_1, \phi_2, \ldots, \phi_r$ compatible $\Rightarrow$ there exists h in H such that $\phi_1, \phi_2, \ldots, \phi_r$ are h-compatible $\Rightarrow$ there exist $h_1^{(i)}$ and $h_2^{(i)}$ such that $\phi_i = h f h_1^{(i)} f h_2^{(i)}$ for i = 1, 2, ... , r.

$h \in H \Rightarrow$ there exists $t : R_n x R_n \rightarrow R_n$ such that $h(pn + q) = pn + t(p,q)$ and $t(p,.)$ is a permutation of $R_n$, for all p, q in $R_n$, after lemma 3.

Thus, (i) is satisfied.

To prove (ii), We reason by contradiction.

If, for some i, $\alpha_i(p,q) = \alpha_i(p',q')$ and $p \neq p'$, and $t(p,q) = t(p',q')$, then

$h_2^{(i)}(\alpha_i(p,q)n + t(p,q)) = h_2^{(i)}(\alpha_i(p',q')n + t(p',q'))$.

After lemma 4, $h_2^{(i)}(\alpha_i(p,q)n + t(p,q)) = \alpha_i(p,q)n + \beta_i(p,q) = \phi_i(pn + q)$

$h_2^{(i)}(\alpha_i(p',q')n + t(p',q')) = \alpha_i(p',q')n + \beta_i(p',q') = \phi_i(p'n + q')$.

Therfore, $\phi_i(pn + q) = \phi_i(p'n + q')$.

$\Rightarrow pn + q = p'n + q' \Rightarrow p = p'$ and $q = q' \Rightarrow p = p'$. Contradiction.

Thus,(ii) must be true.

Q.E.D.

The theorem can be easily mapped into a graph-theoretic problem, namely, n-coloring problem. Let $\phi_1, \phi_2, \ldots, \phi_r$ be r permutations in $S_N$. Let G = (V,E) be the following graph: $V = R_n x R_n$ and $((p,q),(p',q')) \in E$ if

(a) $p = p'$ or

(b) $\alpha_i(p,q) = \alpha_i(p',q')$ for some i.

The theorem can now be termed as follows: $\phi_1, \phi_2, \ldots, \phi_r$ are compatible iff G can be n-colored in such a way that no two adjacent nodes have the same color.

To prove this, let t(p,q) be the color of (p,q). (a) is equivalent to (i) and (b) is equivalent to(ii).

The general n-coloring problem is NP-complete, but it remains open whether these graphs have any peculiarities that open the door to a fast (i.e., polynomial) algorithm to n-color them. Such an algorithm would be of paramount importance.

The characterization theorem will enable us to prove in the following sections that the two families of permutations required by FFT and bitonic sorting are both compatible, opening the way to efficient computation of these two interesting problems on B(n,2).

Before getting to the next sections, it is worthwhile to note that not every two permutations are compatible.This is shown by a counter example using the theorem and its graph formulation.

Take $n = 2$.Thus, $N = 4$, and $R_n = \{0,1\}$. $\phi_1 = (0\ 1\ 2)(3)$ and $\phi_2 = (0)(1\ 2\ 3)$. It can be shown that

$\alpha_1(0,0) = \alpha_1(1,0) = 0$ and $\alpha_2(1,1) = \alpha_2(0,0) = 0$

$\alpha_1(0,1) = \alpha_1(1,1) = 1$ $\alpha_2(0,1) = \alpha_2(1,0) = 1$

The corresponding graph G is depicted in fig . 4. G has a 3-clique, and it cannot therefore be 2-colored. Thus, $\phi_1$ and $\phi_2$ are <u>not</u> compatible.

In the rest of the report, we assume that n is a power of 2.

## V. <u>FAST FOURIER TRANSFORM</u> :

To compute FFT in the way described in [7], two permutations are

needed: Shuffle (S), and exchange (E). However, at the end of the computation, the components of the resulting vector are in bit reversed order. To restore the order, the bit reversal permutation (B) is needed. Thus, the overall family of permutations needed by FFT is S, E and B.

The approach is to express S, E and B as $S(pn + q) = \alpha_S(p,q)n + \beta_S(p,q)$, $E(pn + q) = \alpha_E(p,q)n + \beta_E(p,q)$ and $B(pn + q) = \alpha_B(p,q)n + \beta_B(p,q)$ to compute $\alpha_S$, $\alpha_E$ and $\alpha_B$. Afterwards, we find the conditions (C) that p,q,p',q' should satisfy when $p \neq p'$ in order that $\alpha(p,q) = \alpha(p',q')$ for $\alpha = \alpha_S$, $\alpha_E$ or $\alpha_B$. Then, we see if we can find t such that $t(p,q) \neq t(p',q')$ wherever $(p,q,p',q') \in (C)$.

Before we proceed, the following lemma should prove useful:

Lemma 5: If $\phi_1, \phi_2, \ldots, \phi_r$ are compatible, then the family $H \cup \{\phi_1, \phi_2, \ldots, \phi_r\}$ is compatible too.

Proof: Suppose that $\phi_1, \phi_2, \ldots, \phi_r$ are $h_0$-compatible for some $h_0$ in H. For every h in H, $h = h_0 f e f (h_0^{-1} h)$ because $f = f^{-1}$ (for the $N = n^2$ case). Set column 0 to $h_0$, column 1 to e (identity) and column 2 to $h_0^{-1}h$ which is in H because H is a subgroup. Hence, H is $h_0$-compatible and consequently $H \cup \{\phi_1, \phi_2, \ldots, \phi_r\}$ is a compatible family.

Q.E.D.

E is in H because n is even. Therefore, it is enough to prove that S and B are compatible.

$$S(pn + q) = \begin{cases} 2(pn + q) & \text{if } pn + q \leq N/2 - 1 \\ 2(pn + q) - N + 1 & \text{if } N/2 \leq pn + q \leq N-1 \end{cases}$$

$\Rightarrow$

$$S(pn + q) = \begin{cases} 2pn + 2q & \text{if } p \leq n/2 - 1,\ q \leq n/2 - 1 \quad (1) \\ (2p + 1)n + 2q - n & \text{if } p \leq n/2 - 1,\ q \geq n/2 \quad (2) \end{cases}$$

$$\begin{cases} (2p-n)n + 2q + 1 & \text{if } p \geq n/2,\ q \leq n/2 - 1 \quad (3) \\ ((2p-n+1)n + 2q + 1 - n & \text{if } p \geq n/2,\ q \geq n/2 \quad (4) \end{cases}$$

$\alpha_S(p,q)$ is the coefficient of n in the definition of S.

If $p \neq p'$ (say, $p < p'$) and $\alpha_S(p,q) = \alpha_S(p',q')$, then

$((p,q) \in (1)$ and $(p',q') \in (3))$ or $((p,q) \in (2)$ and $(p',q') \in (4))$ because otherwise $\alpha_S(p,q)$ and $\alpha_S(p',q')$ would be of different parity.

$((p,q) \in (1), (p',q') \in (3)$ and $\alpha_S(p,q) = \alpha_S(p',q')) \Rightarrow 2p = 2p' - n$ and $q,q' \leq n/2 - 1$

$\Rightarrow p' = p + n/2$ and $q,q' \leq n/2 - 1$ and $p \leq n/2 - 1$

$((p,q) \in (2), (p',q') \in (4)$ and $\alpha_S(p,q) = \alpha_S(p',q')) \Rightarrow 2p = 2p' - n$ and $q,q' \geq n/2$

$\Rightarrow p' = p + n/2$ and $q,q' \geq n/2$ and $p \leq n/2 - 1$

$\Rightarrow$ condition (C1) that p, q, p', q' should satisfy is:

(C1) $\equiv p' = p + n/2$ for $p \leq n/2 - 1$ and $((q, q' \leq n/2 - 1)$ or $(q,q' \geq n/2))$

(C1) means that $t(p,q) \neq t(p + n/2,q')$ for $p' = p + n/2$ and $p \leq n/2 - 1$ and $((q, q' \leq n/2 - 1)$ or $(q,q' \geq n/2))$. In terms of switch constraints, (C1) says that for all $p \leq n/2 - 1$, switch p and switch $p + n/2$ of column 0 should be set such that if the first half of switch p maps to some subset k of $R_n$, then the first half of switch $p + n/2$ maps to $R_n$-k.

What remains is to find $\alpha_B$ and (C2) that t should satisfy. Suppose that $n = 2^l$, $N = 2^{2^l}$. $B(a_{2^l-1}a_{2^l-2} \ldots a_0) = a_0a_1 \ldots a_{2^l-1}$ if $a_{2^l-1}a_{2^l-2} \ldots a_0$ is a binary number.

$B(pn + q) = B(q)n + B(p) \Rightarrow \alpha_B(p,q) = B(q)$.

$\alpha_B(p,q) = \alpha_B(p',q') \Rightarrow B(q) = B(q') \Rightarrow q = q'$

Thus, $(C2) \equiv p \neq p'$ and $q = q'$

$\Rightarrow t(p,q) \neq t(p',q)$ for every q and every $p \neq p'$.

That means that no two switches of column 0 are set in such a way that they map the same relative input port to tha same relative output port.

Theorem 2: $t(p,q) = (p + q) \bmod n$ satisfies (C1) and (C2), maps $R_n \times R_n$ to $R_n$ and t(p,.)

is a permutation for every p.

Proof: Clearly $0 \le ((p + q) \bmod n) \le n-1$ for $(p,q) \in R_n x R_n$. Therefore, t maps $R_n x R_n$ to $R_n$.

t satisfies (C1): Fix $p \le n/2 - 1$, and $q, q' \le n/2 - 1$.

If $t(p,q) = t(p + n/2, q')$, then $p + q = (p + n/2 + q') \bmod n$.

$\Rightarrow q = (q' + n/2) \bmod n$.

$q' \le n/2 - 1 \Rightarrow n/2 \le q' + n/2 \le n - 1$, and since $q \le n - 1$, we would have $q = q' + n/2$ implying $q \ge n/2$. Contradiction.

Thus, $t(p,q) \neq t(p + n/2, q')$.

If $q, q' \ge n/2$, it can be similarly shown that

$t(p,q) \neq t(p + n/2, q')$.

t satisfies (C2): Fix p, p' and q such that $p \neq p'$.

If $t(p,q) = t(p',q)$, then $p + q = (p' + q) \bmod n$. Thus, $p = p' \bmod n$ and consequently $p = p'$ because $p, p' \le n-1$. Contradiction.

t(p,.) is a permutation: $t(p,q) = t(p,q') \Rightarrow (p + q) \bmod n = (p + q') \bmod n$

$\Rightarrow q = q' \bmod n \Rightarrow q = q'$.

Q.E.D.

Hence, B and S are h-compatible where $h(pn + q) = pn + t(p,q)$. Finally, B, S and E are all h-compatible and FFT can be computed on B(n,2) with great efficiency: Set column 0 to h at the outset of the computation of FFT, then any permutation needed(i.e., E, B, S) is self-routed.

## VI. **BITONIC SORTING**:

As shown in [7], N numbers can be sorted in $O(\log^2 N)$ time using a

bitonic network of $\log N(\log N + 1)/2$ stages of switches. The switches have comparison power, and the interconnections between the stages are shuffle (S), unshuffle (U), shuffles within segments ($S_i$) and unshuffles within segments ($U_i$) to be defined more rigorously later. Since the switches in B(n,2) have no comparison power, the comparisons done by the "bitonic switches" should be done by the pe's of B(n,2). This can be implemented (or simulated) by an exchange permutation and bitonic switch i is simulated by $pe_{2i}$ and $pe_{2i+1}$ as follows:

Suppose x and y are the inputs to bitonic switch i. By our simulation, x is in $pe_{2i}$ and y in $pe_{2i+1}$. After doing the exchange (via E), both $pe_{2i}$ and $pe_{2i+1}$ have both x and y. If the switch is in state (a) as in fig. 5, $pe_{2i}$ keeps min(x,y) and discards max(x,y), while $pe_{2i+1}$ does the opposite. If in state (b), $pe_{2i}$ keeps max(x,y) and discards min(x,y), while $pe_{2i+1}$ does the opposite. Consequently, the bitonic network is simulated on B(n,2) by $2\log N(\log N + 1)/2 = \log N(\log N + 1)$ permutations (not pairwise distinct). The interesting fact is they are all compatible.

To show their compatibility, we have to define precisely what the shuffles and unshuffles within segments are.

For $0 \le i \le 2^l - 1$ (recall that $n = 2^l$), $R_N$ is broken into $2^i$ contiguous segments, each of length $2^{2^l - i} = N/2^i$. The k-th segment ranges from $kN/2^i$ to $(k+1)N/2^i - 1$. Let $S_i$ be the permutation that shuffles each of the $2^i$ segments as if the segment stands alone, and $U_i$ the permutation that unshuffles each of the $2^i$ segments. The interconnections between the stages of the bitonic network are the $S_i$'s and $U_i$'s. For $i \ge l$, the segments fall inside the switches of B(n,2), making $S_i$ and $U_i$ elements of H. Consequently,it remains to show that $S_0, S_1, \ldots, S_{l-1}, U_0, U_1, \ldots, U_{l-1}$ are compatible. Note that the exchange permutation (E) is in H and therefore does not have to be considered.

Lemma 6: Define $\alpha_i(p,q) = \lfloor S_i(pn+q)/n \rfloor$. Then

$$\alpha_i(p,q) = \begin{cases} 2p - kn/2^i & \text{if } kn/2^i \le p \le (2k+1)n/2^{i+1} - 1 \text{ and } q \le n/2 - 1 \quad (1)_k \\ 2p - kn/2^i + 1 & \text{if } kn/2^i \le p \le (2k+1)n/2^{i+1} - 1 \text{ and } q \ge n/2 \quad (2)_k \\ 2p - (k+1)n/2^i & \text{if } (2k+1)n/2^{i+1} \le p \le (k+1)n/2^i - 1 \text{ and } q \le n/2 - 1 \quad (3)_k \\ 2p - (k+1)n/2^i + 1 & \text{if } (2k+1)n/2^{i+1} \le p \le (k+1)n/2^i - 1 \text{ and } q \ge n/2 \quad (4)_k \end{cases}$$

for $0 \le k \le 2^i - 1$

and

$(\alpha_i(p,q) = \alpha_i(p',q') \text{ and } p < p') \Leftrightarrow (p' = p + n/2^{i+1}$ and $kn/2^i \le p \le (2k+1)n/2^{i+1} - 1$ for some $k$ and $((q, q' \le n/2 - 1) \text{ or } (q, q' \ge n/2)))$

Proof: The proof is nothing more than expressing $S_i(pn+q) = p'n + q'$ for some $p' \le n-1$ and $q' \le n-1$, function of p and q. Then, have $\alpha_i(p,q) = p'$.

$$S_i(pn+q) = \begin{cases} 2(pn + q - kN/2^i) + kN/2^i & \text{if } kN/2^i \le pn + q \le (2k+1)N/2^{i+1} - 1 \\ 2(pn + q - kN/2^i) - N/2^i + kN/2^i + 1 & \text{if } (2k+1)N/2^{i+1} \le pn + q \le (k+1)N/2^i \end{cases}$$

$\Rightarrow$

$$S_i(pn+q) = \begin{cases} (2p - kn/2^i)n + 2q & \text{if } (p,q) \in (1)_k \\ (2p - kn/2^i + 1)n + 2q - n & \text{if } (p,q) \in (2)_k \\ (2p - (k+1)n/2^i)n + 2q + 1 & \text{if } (p,q) \in (3)_k \\ (2p - (k+1)n/2^i + 1)n + 2q + 1 - n & \text{if } (p,q) \in (4)_k \end{cases}$$

It is trivial now to derive $\alpha_i$. It is also trivial to prove that $kn/2^i \le \alpha_i(p,q) \le (k+1)n/2^i - 1$ if (p,q) is in the k-th segment, implying that if $\alpha_i(p,q) = \alpha_i(p',q')$, then both (p,q) and (p',q') are in the k-th segment.

For $i \le l - 1$, $n/2^i = 2^{l-i}$ is even. Hence, $\alpha_i(p,q)$ is even in $(1)_k$ and $(3)_k$ and odd in $(2)_k$ and $(4)_k$.

So we conclude that

$(\alpha_i(p,q) = \alpha_i(p',q')$ and $p<p') \Rightarrow ((\text{for some } k \le 2^i - 1)$

$((p,q) \in (1)_k$ and $(p',q') \in (3)_k)$ or

$((p,q) \in (2)_k$ and $(p',q') \in (4)_k))$

$((p,q) \in (1)_k$ and $(p',q') \in (3)_k) \Rightarrow p' = p + n/2^{i+1}$ & $kn/2^i \le p \le (2k+1)n/2^{i+1} - 1$

& $q,q' \le n/2 - 1$

$((p,q) \in (2)_k$ and $(p',q') \in (4)_k) \Rightarrow p' = p + n/2^{i+1}$ & $kn/2^i \le p \le (2k+1)n/2^{i+1} - 1$

& $q,q' \ge n/2$

Hence

$(\alpha_i(p,q) = \alpha_i(p',q')$ and $p < p') \Rightarrow (p' = p + n/2^{i+1}$

and $kn/2^i \le p \le (2k+1)n/2^{i+1} - 1$ for some $k$

and $((q, q' \le n/2 - 1)$ or $(q,q' \ge n/2)))$

The other direction is simple.

Q.E.D.

Therefore, in the search for $t(p,q)$, $t$ must satisfy condition

(C1) $\equiv t(p,q) \ne t(p + n/2^{i+1}, q')$ for every $k \le 2^i - 1$, and every $i \le l - 1$ and every $p, q, q'$ such that $kn/2^i \le p \le (2k+1)n/2^{i+1} - 1$ and $((q, q' \le n/2 - 1)$ or $(q,q' \ge n/2))$

Next, we will do the same with the $U_i$'s.

Lemma 7: Define $\alpha_i(p,q) = \lfloor U_i(pn + q)/n \rfloor$, then

$$\alpha_i(p,q) = \begin{cases} (p + kn/2^i)/2 & \text{if } kn/2^i \le p \le (k+1)n/2^i - 1, q \text{ and } p \text{ even} \quad (1)_k \\ (p - 1 + kn/2^i)/2 & \text{if } kn/2^i \le p \le (k+1)n/2^i - 1, q \text{ even}, p \text{ odd} \quad (2)_k \\ (p + (k+1)n/2^i)/2 & \text{if } kn/2^i \le p \le (k+1)n/2^i - 1, q \text{ odd}, p \text{ even} \quad (3)_k \\ (p - 1 + (k+1)n/2^i)/2 & \text{if } kn/2^i \le p \le (k+1)n/2^i - 1, q \text{ even}, p \text{ odd} \quad (4)_k \end{cases}$$

for $0 \le k \le 2^i - 1$

and

(for some i)($\alpha_i(p,q) \neq \alpha_i(p,q)$ and $p < p'$) $\Leftrightarrow$ ($p' = p + 1$and p is even

and (q, q' havethe same parity))

Proof: The proof follows the same line of reasoning as the previous lemma.

Therefore, we will only express $U_i(pn + q)$.

$$U_i(pn+q) = \begin{cases} (pn + q - kN/2^i)/2 + kN/2^i & \text{if } kN/2^i \le pn+q \le (k+1)N/2^i - 1, \\ & pn+q \text{ even} \\ (pn + q - kN/2^i\, N/2^i - 1)/2 + kN/2^i & \text{if } kN/2^i \le pn+q \le (k+1)N/2^i - 1, \\ & pn+q \text{ odd} \end{cases}$$

Consequently,

$$U_i(pn+q) = \begin{cases} (p + kn/2^i)n/2 + q/2 & \text{if } (p,q) \in (1)_k \\ (p - 1 + kn/2^i)n/2 + (q+n)/2 & \text{if } (p,q) \in (2)_k \\ (p + (k+1)n/2^i)n/2 + (q-1)/2 & \text{if } (p,q) \in (3)_k \\ (p - 1 + (k+1)n/2^i)n/2 + (q - 1 + n)/2 & \text{if } (p,q) \in (4)_k \end{cases}$$

for $0 \le k \le 2^i - 1$.

The rest is now simple.

Q.E.D.

As a result, t(p,q) must satisfy a second condition:

(C2) $\equiv t(p,q) \neq t(p+1,q')$ if p is even and (q, q' have the same parity)

The question now is whether there is such t that satisfies (C1) and (C2). The answer is affirmative and is given in the next two theorems. We give first two definitions.

Definition 8: If E is a set of numbers and x is a number, E + x is defined to be the set

$\{y + x \mid y \in E\}$.

The next definition divides column 0 into 2 disjoint subsets of switches where t will be defined.

Definition 9: Let $F_1 = \{0\}$, $E_1 = \{1\}$ and recursively $F_i = F_{i-1} \cup (E_{i-1} + 2^{i-1})$ and

$E_i = E_{i-1} \cup (F_{i-1} + 2^{i-1})$.

Theorem 3: (i) $E_l \cap F_l = \varnothing$

(ii) $E_l \cup F_l = R_n$

(iii) $(\forall j \leq n/2 - 1)$(2j is in one of the two sets $E_l$ and $F_l \Rightarrow 2j + 1$ is in the other set)

(iv) $(\forall i = 0, 1, \ldots, l-1)(\forall k \leq 2^i - 1)(kn/2^i \leq p \leq (2k + 1)n/2^{i+1} - 1)$(p is in one of the sets $E_l$, $F_l \Rightarrow p + n/2^{i+1}$ is in the other set)

Proof: By induction on $l = \log n$.

Basis: $l = 1, n = 2, R_n = \{0,1\}$.

$E_1 = \{1\}, F_1 = \{0\} \Rightarrow E_l \cap F_l = \varnothing$ and $E_l \cup F_l = R_n$ proving (i) and (ii).

(iii): $j \leq 2n - 1 \Rightarrow j = 0 \Rightarrow 2j = 0$

$2j = 0 \in F_1 \Rightarrow 2j + 1 = 1 \in E_1$.

(iv): $i = 0, 1, \ldots, l - 1 \Rightarrow i = 0$. We have $k = 0, 0 \leq p \leq 0, p = 0$.

$p = 0 \in F_1 \Rightarrow p + n/2^{i+1} = p + 2/2 = p + 1 \in E_1$

Induction step: Suppose the theorem is true for all values up to

$l - 1 = \log(n/2)$, and we now prove it for the value $l = \log n$.

(i): $E_l \cap F_l = (E_{l-1} \cup (F_{l-1} + 2^{l-1})) \cap (F_{l-1} \cup (E_{l-1} + 2^{i-1}))$

$E_{l-1} \cap F_{l-1} = \varnothing$ by the inductive hypothesis (I.H.)

$\Rightarrow (E_{l-1} + 2^{i-1}) \cap (F_{l-1} + 2^{l-1}) = \varnothing$

$E_{l-1} \cap (E_{l-1} + 2^{i-1}) = \varnothing$ because the elements of $E_{l-1}$ are $\leq 2^{l-1} - 1$ and the elements of $E_{l-1} + 2^{i-1}$ are all $\geq 2^{l-1}$.

Same for $F_{l-1} \cap (F_{l-1} + 2^{i-1}) = \varnothing$

Thus, $E_l \cap F_l = \varnothing$.

(ii): $E_{l-1} \cup F_{l-1} = \{0, 1, \ldots, 2^{l-1}\}$ by (I.H.).

It is easy to see that

$E_l \cup F_l = (E_{l-1} \cup F_{l-1}) \cup ((E_{l-1} \cup F_{l-1}) + 2^{l-1})$

$= \{0, 1, \ldots, 2^{l-1} - 1\} \cup \{2^{l-1}, 2^{l-1}, \ldots, 2^l - 1\}$

$= \{0, 1, \ldots, 2^l - 1\} = R_n$.

(iii): Fix $j \leq n/2 - 1$ and assume $2j \in E_l$.

$E_l = E_{l-1} \cup (F_{l-1} + 2^{l-1}) \Rightarrow 2j \in E_{l-1}$ or $2j \in F_{l-1} + 2^{l-1}$.

If $2j \in E_{l-1}$, then $2j + 1 \in F_{l-1}$ by the (I.H.); hence, $2j + 1 \in F_l$.

If $2j \in F_{l-1} + 2^{l-1}$, then $2j = k + 2^{l-1}$ for some $k \in F_{l-1}$.

k is even. $k/2 = j - 2^{l-2} \leq n/2 - 1 - n/4 = n/4 - 1 = 2^{l-1}/2 - 1$.

$(k/2 \leq 2^{l-1}/2 - 1$ and $2(k/2) = k \in F_{l-1}) \Rightarrow k + 1 \in E_{l-1}$ by the (I.H.).

$2j + 1 = (k + 1) + 2^{l-1} \in E_{l-1} + 2^{l-1} \subseteq F_l \Rightarrow 2j + 1 \in F_l$.

A similar proof applies if $2j \in F_l$.

(iv): Fix $i \leq l - 1$, $k \leq 2^i - 1$, p such that $kn/2^i \leq p \leq (2k + 1)n/2^{i+1} - 1$, and assume $p \in E_l$. We want to prove that $p + n/2^{i+1} \in F_l$.

Sinse $E_l = E_{l-1} \cup (F_{l-1} + 2^{l-1})$, we have two cases: $p \in E_{l-1}$ or $p \in F_{l-1} + 2^{l-1}$.

**Case 1**: $p \in E_{l-1}$

$p \in E_{l-1} \Rightarrow p \leq 2^{l-1} - 1 \Rightarrow kn/2^i \leq 2^{l-1} - 1$ as $n = 2^l$

$\Rightarrow k2^{l-i} < 2^{l-1} \Rightarrow k \leq 2^{i-1} - 1$. Note that

$n/2^i = (n/2)/2^{i-1} = 2^{l-1}/2^{i-1}$. Thus, $k2^{l-1}/2^{i-1} \leq p \leq$

$(2k + 1)2^{l-1}/2^{(i-1)+1} - 1$.

And since $K \leq 2^{i-1} - 1$ and $i - 1 \leq l - 2 = (l - 1) - 1$, we can apply the inductive hypothesis and have

$p + (n/2)/2^{(i-1)+1} \in F_{l-1}$.

Therefore, $p + n/2^{i+1} \in F_{l-1}$, which implies $p + n/2^{i+1} \in F_l$ because $F_{l-1} \subseteq F_l$.

**Case 2**: $P \in F_{l-1} + 2^{l-1}$

$\Rightarrow p = s + 2^{l-1}$ for some s in $F_{l-1}$.

$2^{l-1} \leqq p \Rightarrow 2^{l-1} \leqq (2k+1)n/2^{i+1} - 1 \Rightarrow k \geqq 2^{i-1}$.

Let $k' = k - 2^{i-1}$.

$kn/2^i \leqq p \leqq (2k+1)n/2^{i+1} - 1 \Rightarrow$

$(k - 2^{i-1})n/2^i \leqq s \leqq (2(k - 2^{i-1}) + 1)n/2^{i+1} - 1$ because $n = 2^l$

$\Rightarrow k'n/2^i \leqq s \leqq (2k' + 1)n/2^{i+1} - 1$.

Since $0 \leqq k' \leqq 2^{i-1} - 2^{i-1} = 2^{i-1} - 1$, we are in a situation similar to case 1 above, except $s \in F_{l-1}$ instead of $E_{l-1}$. Due to the symmetry of the problem at hand (symmetry between The E's and the F's), the reasoning leads to $s + n/2^{i+1} \in E_{l-1}$.

$p + n/2^{i+1} = s + n/2^{i+1} + 2^{l-1} \in E_{l-1} + 2^{l-1} \subseteq F_l$

$\Rightarrow p + n/2^{i+1} \in F_l$.

End of case 2.

Therefore, $p + n/2^{i+1} \in F_l$ in either case.

If $p \in F_l$, a similar proof shows that $p + n/2^{i+1} \in E_l$.

Q.E.D.

Theorem 4: If all the switches of column 0 that are in $E_l$ are set to identity (i.e., $t(p,q) = q$ for all p in E and $0 \leq q \leq n - 1$), and all the switches that are in $F_l$ set to r where $r(q) = n - 1 - q$, (i.e., $t(p,q) = r(q)$ for every p in $F_l$), then (C1) and (C2) will be satisfied, and consequently all the permutations required by bitonic sorting are h-compatible where $h(pn + q) = pn + t(p,q)$.

Proof: t is well-defined (no ambiguity) because $E_l \cap F_l = \varnothing$ after (i) of last theorem. t is totally defined because $E_l \cup F_l = R_n$.

(C1) is satisfied: If $kn/2^i \leq p \leq (2k + 1)n/2^{i+1} - 1$ for some i, k and p, then $p \in E_l$ and $p + n/2^{i+1} \in F_l$ or vice versa after (iv) of last thoerem. Assume $p \in E_l$ and $p + n/2^{i+1} \in F_l$. Then, $t(p,q) = q$ and $t(p + n/2^{i+1},q') = n - 1 - q'$. If $t(p,q) = t(p + n/2^{i+1},q')$, then $q = n - 1 - q'$ which makes $q + q' = n - 1$. But if $q, q' \leq n/2 - 1$ then $q + q' \leq n - 2 < n - 1$, and if $q, q' \geq n/2$ then $q + q' \geq n > n - 1$, making it impossible that $q + q' = n - 1$. Thus, $t(p,q) \neq t(p + n/2^{i+1},q')$ if $(p,q,q') \in (C1)$.

(C2) is satisfied: If p is even, then either p is in $E_l$ and $p + 1$ in $F_l$ or vice versa after (iii) of last theorem. Assume p is in $E_l$ and $p + 1$ in $F_l$.

$t(p,q) = q$ and $t(p + 1,q') = r(q') = n - 1 - q' \Rightarrow$ q' and r(q') have different parities $\Rightarrow$ (if q and q' are of equal parity, then q and r(q') have differents parities and therefore cannot be equal).

Thus, $t(p,q) \neq t(p + 1,q')$ if p is even and (q and q' have same parity).

Q.E.D.

## VII. USE OF BITONIC SORTING COMPATIBILITY :

The compatibility of bitonic sorting offers two possibilities concerning interconnection networks. First, a new network can be derived from B(n,2) by replacing the first column by the fixed connection that is the configuration discussed in theorem 4. This network is omega-like (actually it is $h\Omega$ for some h), and consequently self-routed, hardware cheaper, and with full communication power. It can pass any permutation in logN(logN + 1) passes (for $N = n^2$) by sorting the destination tags of the permutation. This is so because the sorting of the tags can be done on this network by logN(logN + 1) self-routed permutations, after last section. Thus, this network is as time efficient as Stone's shuffle-exchange (S-E) network [7]. Although $h\Omega$ costs more hardware than (S-E) net, it has an advantage over (S-E) net in that $h\Omega$ passes many permutations by far fewer passes than (S-E) net does. For instance, the permutations of H are in $h\Omega$ (i.e., passed by one single pass) as shown below. Moreover, $\Omega(n,2)$ and $\Omega^{-1}(n,2)$ are passed in two passes by $h\Omega$. This is proven below.

Lemma 9: $H \subseteq h\Omega(n,2)$.

Proof: For every h' in H, $h' = hfef(h^{-1}h')$ and $h^{-1}h' \in H$ because H is a subgroup.

Q.E.D.

Claim1: $\Omega(n,2)$ is simulated on $h\Omega(n,2)$ in two passes.

Proof: We have to prove that any permutation g in $\Omega(n,2)$ can be written as

$g = g_1g_2$ for some $g_1, g_2$ in $h\Omega(n,2)$.

Fix g in $\Omega(n,2)$. Then $g = fh_1fh_2$ for some $h_1, h_2$ in H because $\Omega = fHfH$.

$g = h^{-1}(hfh_1fh_2)$. Let $g_1 = hfh_1fh_2$ which is in $h\Omega(n,2)$,

and $g_2 = h^{-1} \in H \subseteq h\Omega(n,2)$.

Hence $g = g_1g_2$ where both $g_1$ and $g_2$ are in $h\Omega(n,2)$.

To pass g, $h\Omega(n,2)$ passes $g_1$ then $g_2$.

Q.E.D.

Claim 2: $\Omega^{-1}(n,2)$ is simulated on $h\Omega(n,2)$ in two passes.

Proof: $\Omega^{-1}(n,2) = HfHf$. Fix g in $\Omega^{-1}$. Then $g = h_0fh_1f$.

$\Rightarrow g = (h_0h^{-1})(hfh_1fe)$. Let $g_1 = h_0h^{-1}$ and $g_2 = hfh_1fe$.

Clearly, $g_2$ is in $h\Omega(n,2)$ and $g_1 \in H \subseteq h\Omega(n,2)$, yielding that $g = g_1g_2$ where both $g_1$ and $g_2$ are in $h\Omega(n,2)$.

Q.E.D.

The second possibility that bitonic sorting compatibility offers is to keep all the three columns of B(n,2). B(n,2) enjoys all the properties of $h\Omega(n,2)$ (except that B(n,2) costs more hardware). Moreover, B(n,2) simulates $\Omega$ in one pass with self-routedness, allows compatible families to be self-routed, and for the incompatible families, B(n,2) can be controlled in O(logN(logN + 1)) time using bitonic sorting to sort the destination tags.

## VIII. SIMULATION OF FIXED (IN)'s ON B(N,2):

In this section, it will be proven that many of the standard fixed interconnection networks can be simulated on B(n,2) by compatible families of permutations. Furthermore, the simulation is optimal. We start first by defining precisely what we mean by optimal simulation.

Let T be a fixed interconnection network. A communication step (CS) of T is a data routing process in which one or more pe's send data to their immediate neighbors only (the immediate neighbors of a pe are the pe's to which it has direct links). A full communication step (FCS) of T is a (CS) of T where *every* link is used exactly once (each pe sends data to *all* its immediate neighbors). A communication process is carried out by a number of communication steps. An elementery communication step (ECS) is a (CS) in which each pe sends and/or receives at most one data items. It is clear that a (CS) can be carried out by a sequence of (ECS)'s

although this is not the most efficient way of communication in T. Therefore, any communication process in T can be done by a sequence of (ECS)'s.

From here on, it is assumed that the number of pe's in T is $N = n^2$ (i.e., the same number of those of B(N,2)).

An (ECS) can be done (or simulated) on B(n,2) by means of one permutation $\pi$ in the most natural way, that is, $\pi(i) = j$ if $pe_i$ sends a data item to $pe_j$ in that (ECS). Note that $\pi$ does not have to be totally defined. Hence, any (CS) of T can be done (or simulated) on B(n,2) in a number of permutations equal to the number of its (ECS)'s. We are now in a position to define optimal simulation of T on B(n,2).

Definition 10: The optimal simulation of a fixed (IN) T on B(n,2) is the determination of a minimum set of permutations needed to simulate a full communication step of T on B(n,2).

This last definition can be cast even more formally as follows: A set S of permutations simulates T if for every link $(pe_i,pe_j)$ of T, there is at least one permutation $\pi$ in S such that $\pi(i) = j$. A minimum such set optimally simulates T.

T can be modeled by a directed graph G where the nodes are the pe's and the arcs are the directed links between the pe's. Let the nodes be numbered 0,1, ... , N-1. Let $ind_i$ and $outd_i$ be the indegree and outdegree of node i, respectively.
Let $ind = \max\{ind_i \mid i = 0,1, \ldots ,N-1\}$, $outd = \{outd_i \mid i = 0,1, \ldots ,N-1\}$ and
$d = \max(ind,outd)$. By the definition of a (FCS), $pe_i$ sends $outd_i$ items and receives $ind_i$ items in a (FCS). d is the maximum number of items sent or received by any one pe; therefore, d is a lower bound on number of permutations needed to simulate a (FCS) because a pe can send and/or receive no more than one data item through one permutation.

Consequently, if a set of permutations simulates a (FCS) and is of order d,

it is necessarily a minimum set.

Note that many (IN)'s are presented as undirected graphs where each link represents either (1) a bidirectional link or (2) two opposite links. If a set of permutations simulates T as conceived in (2), it simulates T as conceived in (1). Thus, we adopt (2), that is, an undirected edge represents two distinct links of opposite directions between two pe's. In the rest of the report, if we have an undirected graph G, we mean by directed graph G the graph as conceived in (2).

In the following, hypercubes, four-neighbor nets, six-neighbor nets and eight-neighbor nets will be simulated on B(n,2) in an optimal way.

Definition 11: $M_1, M_2, \ldots, M_k$ are said to be a perfect matching partition of a graph $G = (V,E)$ if each $M_i$ is a perfect matching of G (i.e., $M_i \subseteq E$ and each node of V is incident upon exactly one edge of $M_i$) and $M_1, M_2, \ldots, M_k$ are a partition of E (i.e., $M_1 \cup M_2 \cup \ldots \cup M_k = E$ and $M_i \cap M_j = \varnothing$ $\forall i \neq j$).

Note that if G has a perfect matching partition $M_1, M_2, \ldots, M_k$, then G is regular and the degree of each node is k.

Lemma 8: If a graph G, modelling an (IN) T, has a perfect matching partition $M_1, M_2, \ldots, M_k$, then T is optimally simulated on B(n,2) with k permutations $\pi_1, \pi_2, \ldots, \pi_k$. Moreover each $\pi_i$ is solely derived from $M_i$. We denote $\pi_i$ by $\pi(M_i)$.

Proof: Define $\pi_i$ to be: $\pi_i(j) = k$ iff (j,k) is in $M_i$. Note that if $\pi_i(j) = k$, then $\pi_i(k) = j$. Since $M_i$ is a perfect matching, $\pi_i$ is a permutation. For every link (j,k) in T (i.e., for every directed edge (j,k)), there exists i such that (j,k) is in $M_i$ because $M_1 \cup M_2 \cup \ldots \cup M_k = E$. Thus, there exists $\pi_i$ such that $\pi_i(j) = k$. Hence, $\pi_1, \pi_2, \ldots, \pi_k$ simulate T (after the remark following

definition 10). The simulation is optimal because k is the indegree and the outdegree of each node of directed G and therefore k is a lower bound on the order of simulating sets of permutations.

Q.E.D.

(a) Simulation of Hypercubes:

Assume $N = n^2 = 2^{2k}$, where $n = 2^k$. The hypercube has N pe's. Each pe is identified by a 2k-bit binary number $x_{2k-1} \ldots x_1 x_0$. Two pe's are connected (by two links of opposite directions) if their id's differ in only one bit. The directed hypercube is regular, each $ind_i$ and $outd_i$ is equal to 2k.

Let $M_i = \{(x, x + 2^{i-1}) \mid x = x_{2k-1} \ldots x_i x_{i-1} x_{i-2} \ldots x_1 x_0$ and $x_{i-1} = 0\}$ for $i = 1, 2, \ldots, 2k$. More informally, (x,y) is in $M_i$ iff x and y differ only in the i-th bit (from the left). It is straightfotward to verify that each $M_i$ is a perfect matching and that $M_1, M_2, \ldots, M_{2k}$ form a perfect matchimg partition of the undirected hypercube. After lemma 8, $\pi(M_1), \ldots, \pi(M_{2k})$ simulate the hypercube optimally. What remains is to find each $\pi(M_i)$ and prove that the $\pi$'s are in $\Omega$.

CASE 1: $i \leq k$.

We claim that $\pi(M_i)$ is in H.

Proof: Let (x,y) be an element of $M_i$. $\pi(M_i)$ maps x to y (and viceversa).

$x = pn + q$ and $y = p'n + q'$ for some p,q,p',q'. q contributes the first k bits of x, and p its second k bits. Same for q', p', y.

x and y differ only in the i-th bit and $i \leq k \Rightarrow p = p'$

Thus, $\pi(M_i)$ is in H and consequently $\pi(M_i)$ is in $\Omega(n,2)$, that is, can be passed through B(n,2) with column 0 set to identity.

CASE 2: $k < i \leq 2k$.

$i = k + s$ for some $s \leq k$.

Let (x,y) be as CASE 1. $x = pn + q$ and $y = p'n + q'$.

x and y differ only in the i-th bit and $i > k \Rightarrow (q = q')$ and (p and p' differ only

in the s-th bit)

We thus have that $\pi(M_i)$ maps $pn + q$ to $p'n + q$ preserving q and having p' function of p only (as opposed to p and q). The fact that $\pi(M_i)$ is in $\Omega$ follows from the following lemma.

Lemma 9: If a permutation $\pi$ maps $pn + q$ to $p'n + q$ preserving q and having p' dependent upon p only, then $\pi$ is admissible by B(n,2) with column 0 set to identity.

Proof: Take $h_0 = h_2 = e$ and $h_1$ such that $h_1(qn + p) = qn + p'$ where p' is such that $\pi(pn + q) = p'n + q$. Clearly, $h_1$ is well defined and in H.

It is easy to verify that $\pi = h_0 f h_1 f h_2$. Thus, $\pi$ is admissible by B(n,2) with column 0 set to identity.

Q.E.D.

Consequently, $\pi(M_1), \ldots, \pi(M_{2k})$ are all admissible by $\Omega(n,2)$.

(b) Simulation of Four-Neighbor Nets:

These nets have $N = n^2$ pe's. We assume that n is even. Each pe is connected to its four neighbors (left, right, north, south). Assume that the pe's are numbered by row from left to right, top to bottom, from 0 to N - 1. The $pn + q$-th pe is in the p-th row and q-th column for $0 \leq p,q \leq n - 1$.

After a little inspection, it can be seen that $pn + q$ is connected to $pn + q + 1$, $pn + q - 1$, $(p + 1)n + q$ and $(p - 1)n + q$ if $1 \leq p,q \leq n - 2$. Also, pn is connected to $pn + n - 1$ for $0 \leq p \leq n - 1$; and q is connected to $(n - 1)n + q$ for $0 \leq q \leq n - 1$ (those are the wrap around connections). For the $N = 16$ case, see fig. 7.

Let $M_1 = \{(pn + 2j , pn + 2j + 1) \mid 0 \leq p \leq n - 1, 0 \leq j \leq n/2 - 1\}$

$M2 = \{(pn + 2j + 1, pn + 2j + 2) , (pn , pn + n - 1) \mid 0 \leq p \leq n - 1, 0 \leq j \leq n/2 - 2\}$

$M_3 = \{(2jn + q , (2j + 1)n + q) \mid 0 \leq q \leq n - 1, 0 \leq j \leq n/2 - 1\}$

$M2 = \{((2j + 1)n + q , (2j + 2)n + q) , (q , (n - 1)n + q) \mid 0 \leq p \leq n - 1, 0 \leq j \leq n/2 - 2\}$

It is a trivial matter to see that $M_1$, $M_2$, $M_3$, $M_4$ form a perfect matching partition of the net. The corresponding permutations will be found and proven to be in $\Omega(n,2)$ after giving the definition of some switch states.

Definition 12: 1) An nxn switch (for even n) is in state x (exchange) if input port i is connected to output port i + 1 and input port i + 1 to output port i for i even and $0 \leq i \leq n - 1$.

2) It is in state mx (modified exchange) if input 0 is connected to output n - 1, input n - 1 to output 0, input i to output i + 1 and input i + 1 to output i for i odd and $1 \leq i \leq n - 2$.

3) It is in state i-c, where i is a number, if input j is connected to output $(j + i) \bmod n$ for $0 \leq j \leq n - 1$. This is the i-th cyclic shift.

The four permutations corresponding to $M_1$, $M_2$, $M_3$ and $M_4$ can now be stated in terms of switch configuration:

perm 1: column 0 and 1 set to e, the switches of column 2 all set to x.

perm 2: column 0 and 1 set to e, the switches of column 2 all set to mx.

perm 3: column 0 and 2 set to e, the switches of column 1 all set to x.

perm 4: column 0 and 2 set to e, the switches of column 1 all set to mx.

Clearly, the four permutations are in $\Omega(n,2)$. Needless to say that the simulation is optimal since the net is regular and of degree four.

(c) Simulation of Eight-Neighbor Nets:

The lower bound on the number of permutations is eight because these nets have nodes of degree eight (see fig.9).

We can follow the same line of resoning as in (a) and (b). So we will state the results without going through the same depth of detail.

These nets are optimally simulated by the following eight permutations:

perm1, perm2, perm3 and perm4 are the same as in (b).

perm 5: column 0 set to e, the switches of column 1 and 2 all set to x.

perm 6: column 0 set to e, the switches of column 1 and 2 all set to mx.

perm 7: column 0 set to e, the switches of column 1 all set to x, of column 2 to mx.

perm 8: column 0 set to e, the switches of column 1 all set to mx, of column 2 to x.

(d) Simulation of Six-Neighbor Nets:

The lower bound on the number of permutations is six because these nets have nodes of degree six (see fig. 8).

The following six permutations provide the optimal simulation:

perm1, perm2, perm3 and perm4 are the same as in (b).

For perm 5 and perm 6, we have to divide the third column into four subsets of switches: $E_0$, $E_1$, $E_2$ and $E_3$ where $E_i$ is the set of switches whose indices are congruent to i mod 4 (recall that there are n switches in a column, numbered 0,1, ... , n - 1 from top to bottom).

perm 5: column 0 set to e, the switches of column 1 all set to x, the switches of $E_0$ and $E_2$ of column 2 set to 3-c, the switches of $E_1$ and $E_3$ set to 1-c.

perm 6: column 0 set to e, the switches of column 1 all set to mx, the switches of $E_0$ and $E_1$ of column 2 set to 3-c, the switches of $E_2$ and $E_3$ set to 1-c.

All the six permutations are admissible by $\Omega(n,2)$.

(e) Simulation of Shuffle-Exchange Nets:

After section V on FFT, the shuffle-exchange net is simulated by two permutations : S and E. S is not admissible by $\Omega(n,2)$. However, S and E are h-compatible for some h (see section V).

## IX. GENERALIZATIONS AND OPEN QUESTIONS:

Of interest is whether we can generalize the results of this report concerning B(n,2) to B(n,k) for k>2. some of the definitions have to be generalized.

Generalization of Compatibility: Let $h_0, h_1, \ldots h_{k-2}$ be in H. A permutation $\pi$ is said ( $h_0, h_1, \ldots h_{k-2}$)-compatible if the network resulting from setting column 0, column 1, ... , column k-2 of B(n,k) to $h_0, h_1, \ldots h_{k-2}$ , respectively, passes $\pi$ without conflict. $\pi$ is then said to be compatible in the general sense.

The simulation of the fixed (IN)'s of section VIII gives rise to families of permutations in $\Omega(n,2)$. If $n^2 = r^k$ (k even), then $\Omega(n,2) \subseteq \Omega(r,k)$. Therefore, these families are compatible in the general sense.

The following questions remain open and subject to further pursuit:

1) Are FFT and bitonic sorting compatible in the general sense ?

2) Is there an efficient algorithm to decide compatibility in the original sense ? In the general sense ?

It is also of interest to study other interesting classes of problems and see if they produce compatible families of permutations ?

## X. CONCLUSION:

We have established an approach to controlling Benes networks for the $N = n^2$ case, and showed that this approach is more versatile than other previous approaches. We have also shown how to attack families of permutations (generated by classes of problems) to see if they are compatible, using the characterization theorem. FFT and bitonic sorting were proven to produce compatible families, yielding the way to very efficient communication for these two problems. Standard fixed interconnection networks were simulated on our scheme in an optimal way. There are ather classes of problems that yield to our scheme, too. Such are matrix computations (matrix multiplication, linear transformation, LU decomposition),

semi-group computation(addition and multiplication of N numbers, maximum and minimum of N values), recurrent equations of order $\geq 1$, etc. ... Actually, most of these problems yield families of permutations that are admissible by omega networks and therefore by our scheme, for omega nets are a special case. It is of interest to inspect other classes of problems and see whether they produce families of permutations that can join the "club" of compatible families.

It remains to say a few words about the implementation of the scheme. The implementation can be integrated into the instruction set of the system. For each known h-compatible family, we store the configuration h in memory. A new instruction is added to the instruction set. This instruction takes an operand that points to the location of h. The execution of this instruction consists of setting column 0 to h. If we assume that the system has already an instruction LOAD-PERM that takes a permutation (or a pointer to it) as an operand and sets the network to it, then we can add a bit c to this instruction. If c is set to 0, the same execution takes place; and if c is set to 1, then the net is self-routed using the destination tags (as explained in fact 2 of section II).

This is one of the possible implementations. It shows the ease, elegance and efficiency of the scheme. Another way of using the scheme is to use the bitonic sorting algorithm to sort the destination tags, yielding an $O(\log^2 N)$ control algorithm of B(n,2) , which is as efficient as Stone's S-E net control algorithm, while keepimg the generality and versatility of B(n,2) for compatible families.

## REFERENCES:

[1] K. E. Batcher, "Sorting networks and their applications," Proc. Spring Joint Computer Conf., AFIPS Conf. (Montvale, N.J. :AFIPS Press, 1968), vol. 32, pp. 307-314.

[2] V. E. Benes, "Mathematical theory of connecting networks and telephone traffic," Academic Press, New York, 1935.

[3] M. J. Flynn, "Very high speed computing systems," Proc. IEEE, vol. 54, pp. 1901-1909, 1966.

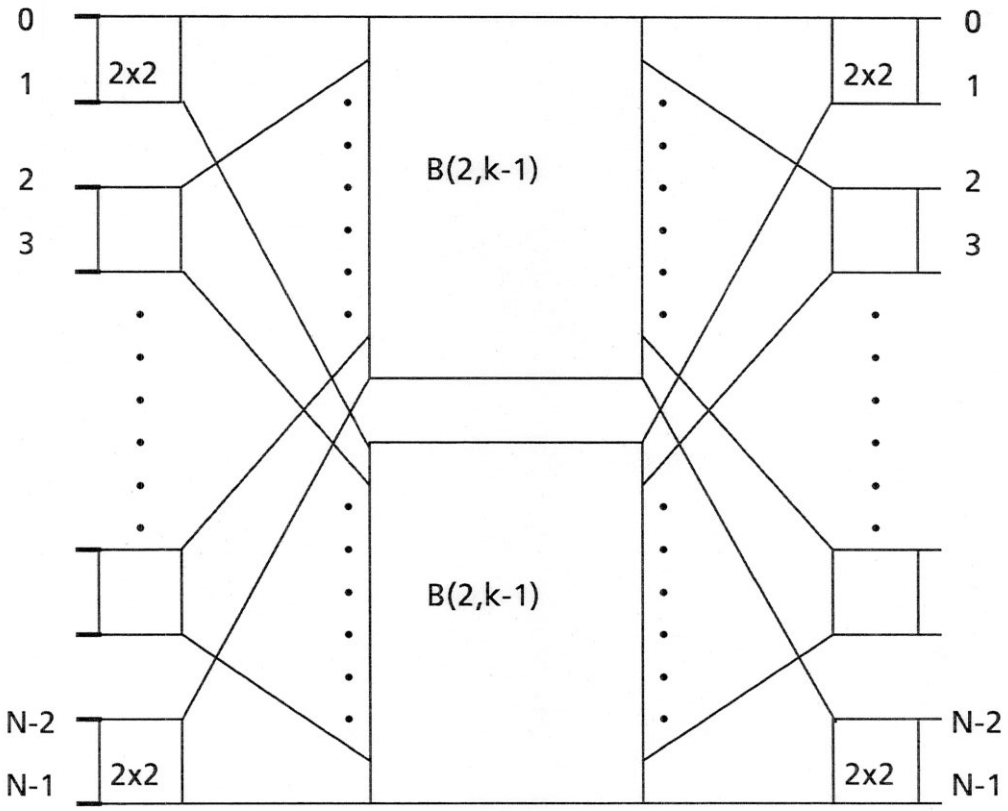
[4] J. Lenfant, "Parallel permutations of data: A Benes network control algorithm for frequently used permutations," IEEE Trans. Comput., vol. C-27, No. 7, pp. 637-647, Jul 1978.

[5] D. Nassimi and S. Sahni, "Parallel algorithms to set-up the Benes permutation network," Univ. of Minnesota, minneapolis, Tech. Rep. 79-19, June 1979; also in proc. Workshop on Interconnection Networks for Parallel and Distributed Processing, Purdue University, Lafayette, IN, Apr. 1980.

[6] D. Nassimi and S. Sahni, "A self-rouing Benes network and parallel permutation algorithms," IEEE Trans. Comput., vol. C-30, No. 5, pp. 332-340, May 1981.

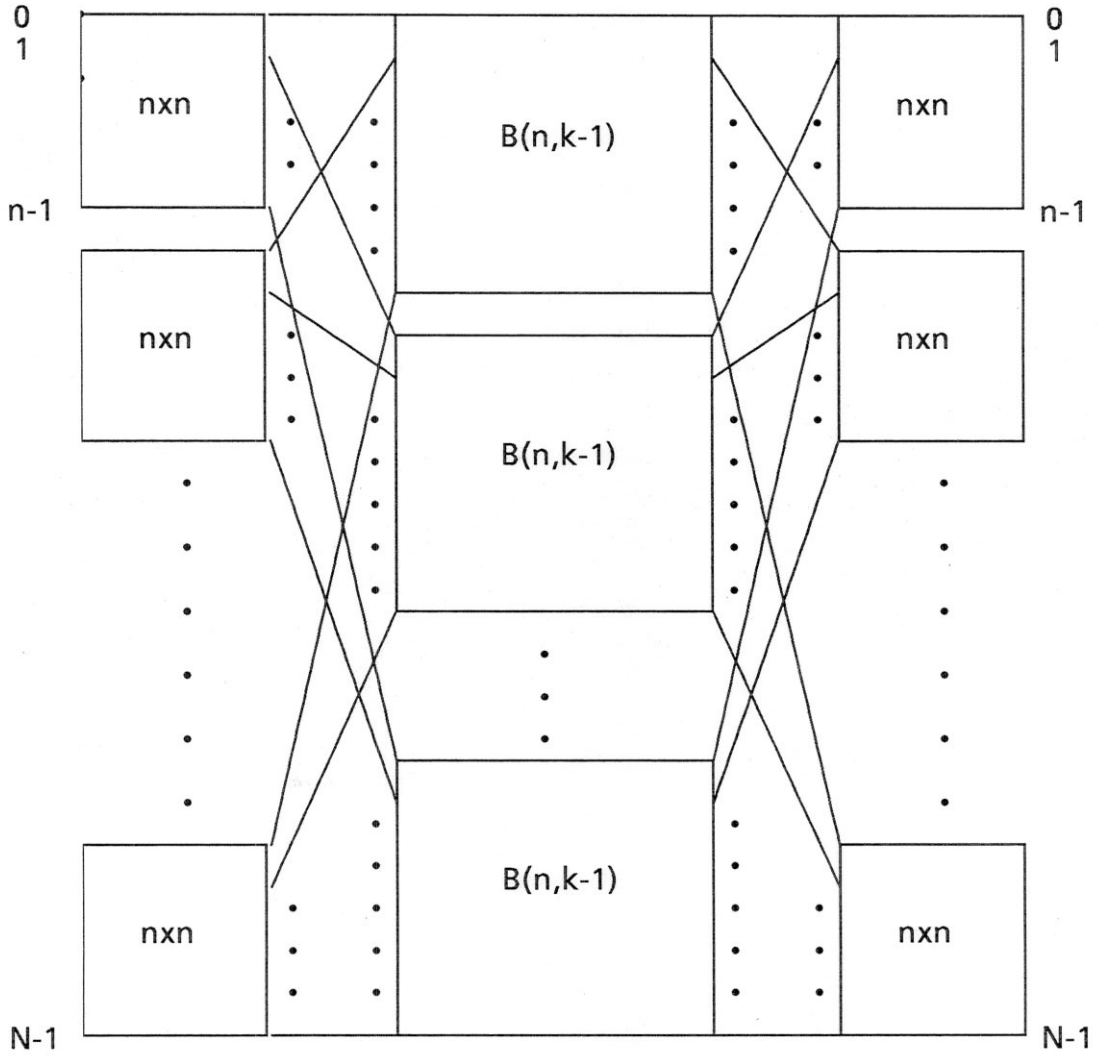
[7] H. S. Stone, "Parallel processing with the perfect shuffle," IEEE Trans. Comput., vol. C-20, pp. 153-161, Feb. 1971.

[8] A. Waksman, "A permutation network," JACM, vol. 15, No. 1, pp. 159-163, Jan 1968.



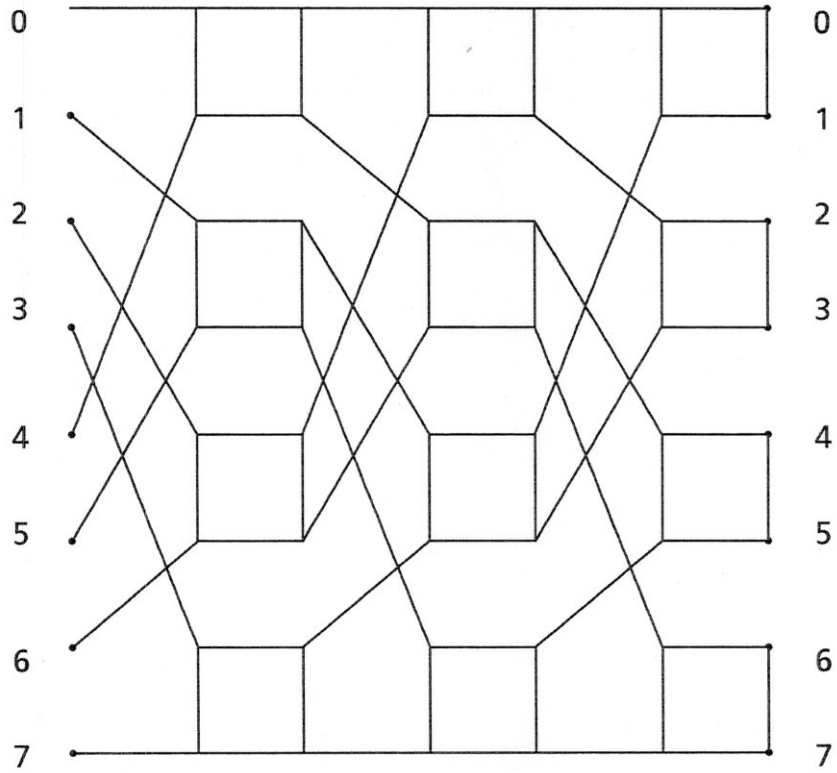

An NxN Benes network (B(2,k), where $N = 2^k$).

Fig. 1




An NxN Benes network, denoted B(n,k) where $N = n^k$. The building block is an nxn switch. Each of the first and last column has N/n switches, while the middle column has n copies of B(n,k-1).

Fig. 2




An 8x8 omega network ($\Omega(2,3)$).

Fig. 3

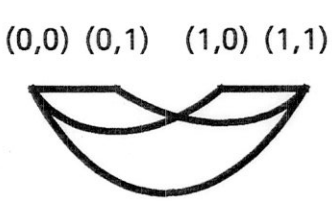
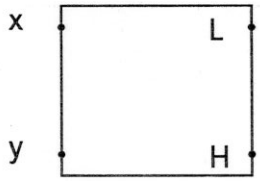
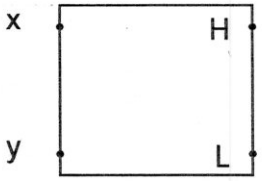



fig. 4

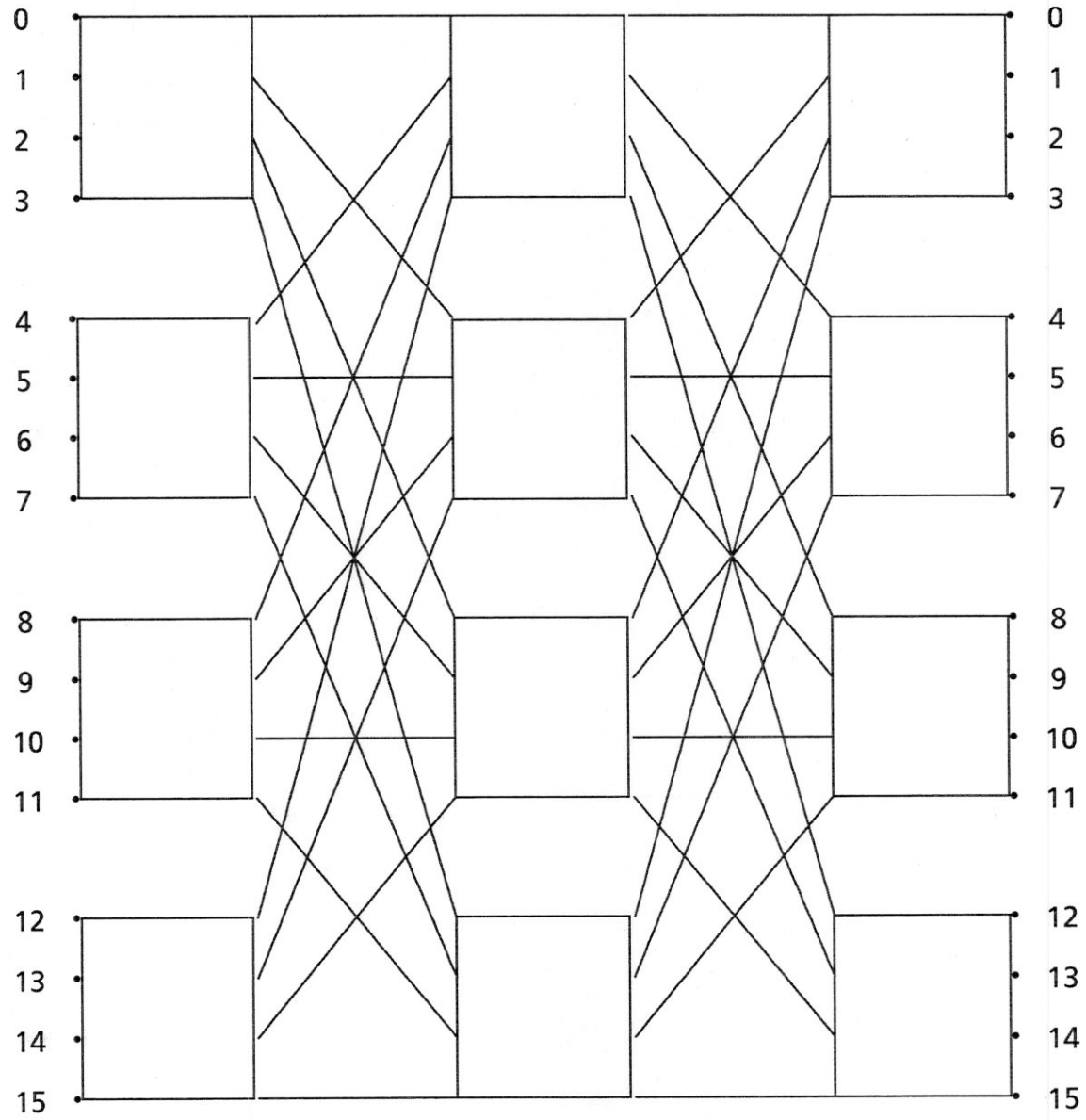



(a)




(b)

fig. 5




A 16x16 Benes network. The building block is a 4x4 switch.

Fig. 6

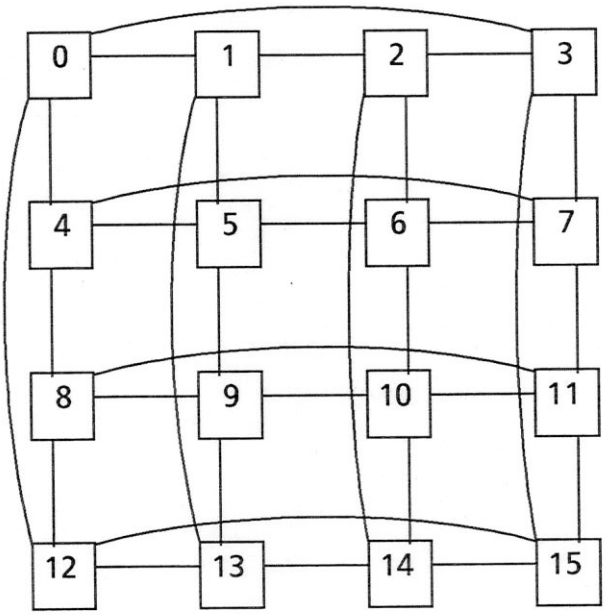



fig.7
Four-neighbor net of 16 pe's

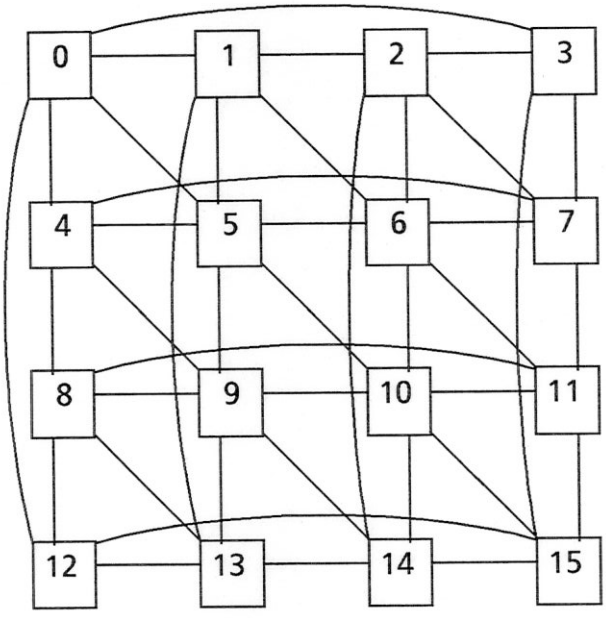



fig.8
Six-neighbor net of 16 pe's

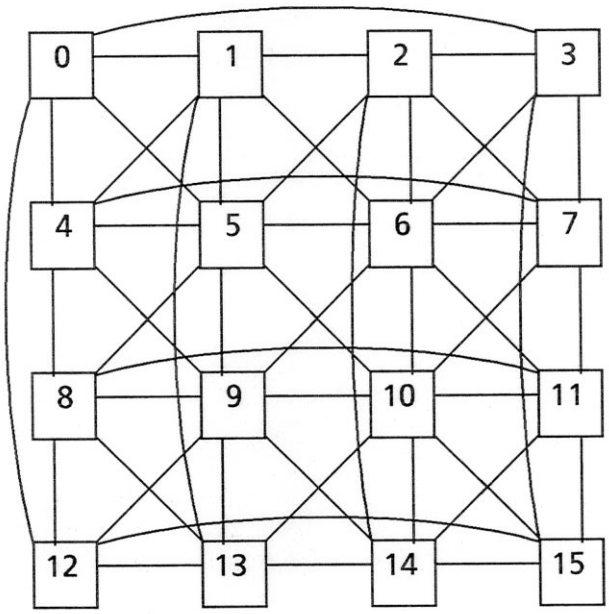



fig.9
Eight-neighbor net of 16 pe's