Report Number: WUCS-81-02

1981-04-01

PIN Limitations and VLSI Interconnection Networks

Mark A. Franklin and Donald F. Wann

Multiple processor interconnection networks can be characterized as having N' inputs and N'
outputs, each B' bits wide. Construction of large networks requires partitioning of the N'*N'*B'
network into a collection of N*N switch modules of data size B (B

Recommended Citation

Franklin, Mark A. and Wann, Donald F., "PIN Limitations and VLSI Interconnection Networks" Report
Number: WUCS-81-02 (1981). *All Computer Science and Engineering Research.*
https://openscholarship.wustl.edu/cse_research/884

PIN Limitations and VLSI Interconnection Networks

Mark A. Franklin and Donald F. Wann

WUCS-81-02

April 1981

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

ABSTRACT

Multiple processor interconnection networks can be characterized as having N' inputs and N' outputs, each B' bits wide. Construction of large networks requires partitioning of the N'*N'*B' network into a collection of N*N switch modules of data size B (B < B') each implemented on a single chip and interconnecting them with a specific interchip network type T'. The major constraint in the VLSI environment is the pin limitation, N_p, of the individual modules; these are allocated between data and control lines, Q. This paper presents a methodology for selecting the optimum values for N and B given values of the parameters, N', B', T', and Q, and N_p. Models for both the banyan and crossbar networks (T') are developed and arrangements yielding minimum: a) number of chips (e.g., switch modules), b) average delay through the network (e.g., maximum bandwidth), and c) product of number of chips and delay, are presented. The results show that for the crossbar a bit slice approach (B = 1) produces the optimum arrangement, while for the banyan the optimum is achieved with multiple bits per module.

PIN LIMITATIONS AND VLSI INTERCONNECTION NETWORKS

Mark A. Franklin and Donald F. Wann
Washington University
St. Louis, Missouri 63130

1. INTRODUCTION

Over the past few years a variety of multiple processor systems have been proposed. Some have even been built while others are now being implemented (1,2,3,4). VLSI technology progress and prospects are now leading computer architects to consider physically local, closely coupled systems in which thousands of processors are present, all contributing to the solution of a given problem(s).

One key issue in the design of such systems concerns the communications network used by these multiprocessor systems. While much of the original work on the design of such networks was done in the context of telephone switching systems (5), current interest in the broader computer community stems largely from the impetus towards multiprocessor systems (6,7,8,9). Recent studies have focused on the functional properties of various networks (i.e., what permutations are possible, what control algorithms are needed, etc.), on their complexity, and to some extent on various performance issues. In most cases network complexity has been measured by the number of elementary switching components needed by a network of a given size and type (e.g., crossbar network complexity grows as $O(N**2)$), while performance (e.g., delay and bandwidth) has been determined by parameters such as the average number of elementary switching components through which a message must pass. Recently work has begun on examining complexity and performance questions in the context of VLSI implementation of such interconnection networks. Franklin (10) has compared two networks,

crossbar and banyan, operating in a circuit switched mode in terms of their space (area) and time (delay) requirements. The networks were assumed to be implemented as complete modules on a single VLSI chip.

Closer examination of network implementation problems shows that pin limitations, rather than chip area or logical component limitations, are a major constraint in designing very large interconnection networks. Consider, for instance, an interconnection network with N' inputs, M' outputs and with each input or output being B' bits wide ($N'*M'*B'$). The number of required pin connections (ignoring power, ground and general control) when the network is implemented on a single chip is given by $B'(N'+M')$. For a square interconnection network of size twelve with a data pathway of 16 bits the number of pins required would thus be 384; much larger than common commercially available integrated circuit carriers. The total number of pins is limited mainly by the increase in the physical length of the package. If the package is to be inserted in pads on a printed circuit board, then the pins are typically placed on 100 mil centers along the periphery (we ignore here certain more advanced schemes such as the array configuration used by IBM). For this pin placement, a 64 pin dual-in-line package is 3.2 inches in length. For the 384 pin example, a 19.2 inch dual-in-line package would be required.

There are a number of potential solutions to this pin limitation problem. In this paper we focus on two of the more obvious ones, investigate their interaction, and develop certain expressions which should aid in their implementation. The first approach is to implement a large network requiring many pins as a collection of smaller networks where each of the smaller networks can be contained on a single chip in which the pin constraints of the chip are

met. An N'*N' network would therefore be decomposed into a set of networks (each network in the set is a chip of size N*N) which would themselves be interconnected in some fashion.

The second approach is to slice the network so that one creates a set of network planes, each plane handling one or more bits (e.g., B bits) of the B' wide datapath. This is commonly done for instance in memory designs. Note that a potential problem arises when doing this due to the difficulty in synchronizing the multiple planes. This can occur even if the network components operate in an asynchronous manner under local control. Although not discussed here, there are ways of dealing with this problem.

The remainder of this paper deals with the question of what represents the "best" combination of datapath slice B and chip network size N given:

1. N': An overall network size ($N <= N'$),

2. B': A data path width ($B <= B'$),

3. T : An intrachip network type (e.g., the interconnection network implemented within the N*N chip might be a crossbar).

4. T': An interchip network type (e.g., the interconnection network implemented between the N*N chips to achieve the overall N'*N' network might be an omega network).

5. N_p: The maximum number of pins allowed on a chip.

6. N_k: The number of pins on a chip allocated to power, ground, and control. Depending on the control scheme adopted, the number of control pins may be a function of N.

"Best" in this context, refers to both chip count and bandwidth of the overall N'*N' network. Figure 1 shows an overall N'*N' network. Figures 2 and 3 illustrate a possible decomposition of a 16*16 network having a datapath width of 8, into 32 4*4 chip networks each having datapath widths equal to 2.

In the next section basic models for considering this problem are presented.

These models are used to determine the B and N combinations which minimize first the total chip count, and then the overall network delay. Various expressions are developed and evaluated for a variety of system parameters, and a number of illustrative graphs are presented. Finally, an overall performance measure involving the product of chip count and time delay is defined and selection of the "best" B and N values for a given set of sample system parameters is presented.

2. THE BASIC MODEL

The basic model consists of two parts. The first relates to the chip count while the second concerns network time delay. For the purposes of this paper the networks are assumed to be square, although the approach presented holds for nonsquare networks as well. It is also assumed that our interest is restricted to fully connected networks where there is a path through the network from each input port to each output port. Note that certain input/output paths may have a common subpath and this may in turn result in messages being temporarily blocked.

Let us refer to the N*N*B chip as a <u>switch module</u>; a number of these modules will be interconnected to realize the N' network. This paper considers two types of interchip networks (T'): the incremental crossbar and the banyan (11,12,13). While there are many ways of designing a crossbar network (e.g., demultiplexer/multiplexer configuration, switched multiple busses, etc), the incremental crossbar design (Figure 4) has the property that it can be expanded on a unit basis by adding basic switch modules in a row-column arrangement. This modularity property is useful if a flexible expansion capability which retains the nonblocking and full connection properties of the crossbar are required. Note that a price is paid for these nonblocking and modularity properties in

terms of number of switches and number of pins required on a switch module. While the number of switches required per switch module may not be a serious constraint given the logic density capabilities of the VLSI technology, as pointed out, the problem of pin constraints is severe. For the incremental crossbar, the modularity property requires 4NB data pins to implement a N*N*B switch module. The banyan, a blocking network, on the other hand requires 2NB data pins to implement a N*N*B module.

To make global comparisons similar and to eliminate blocking at the switch module level, this paper examines cases in which the switch modules are constructed using an incremental crossbar architecture. Two types of module interconnections are examined; the crossbar and the banyan. Therefore:

$$T \Rightarrow \text{Incremental Crossbar}$$

$$T' \Rightarrow \begin{cases} \text{I.} & \text{Incremental Crossbar} \\ \text{II.} & \text{Banyan} \end{cases}$$

2.1 CHIP COUNT MODEL

Figure 4 shows an 8*8*1 incremental crossbar network configured using 2*2*1 chip components. The number of N*N*B chips required to implement an N'*N'*B' incremental crossbar network is given by:

$$N_{cb} = \left\lceil \frac{B'}{B} \right\rceil \left\lceil \frac{N'}{N} \right\rceil^2 \qquad [1]$$

The second type of network considered is one of the class of networks whose logical component complexity grows as O(N log N) rather than O(N**2). The banyan, omega and inverse binary N-cube networks fall into this group. Figure 5 shows an 8*8*1 network composed in a banyan configuration using 2*2*1

network chip components. Note that this network is a blocking network. The number of N*N*B chips needed to implement an N'*N'*B' network in this case is given by:

$$N_{ba} = \left\lceil \frac{B'}{B} \right\rceil \left\lceil \frac{N'}{N} \right\rceil \left\lceil \log_N N' \right\rceil \qquad [2]$$

The first term in this expression is the number of bit slices or network planes that are required. The second term represents the number of chips at each level (row), while the third term is the number of levels necessary to achieve a full interconnection.

2.2 TIME DELAY MODEL

The model that gives the average time for a signal to propagate through a network must include the time to traverse each of the N*N*B chips, the time to propagate from chip to chip, and since the bit slice approach separates the bits in a data word, the additional time that is needed to make certain that all the data bits have completed their movement through the N'*N'*B' network.

The average delay associated with a basic switch module will be designated as D_{CB} since these modules have a crossbar construction. This module delay includes the path (e.g., multiplexing delays) but does not include path setup delays (i.e., time to set switches in their desired positions). The delay of a pin driver and the delay associated with the interconnection wires between modules will be referred to as the intermodule delay, D_{IM}. The intermodule delays for the crossbar and banyan networks are different and will be denoted as D_{IMCB} and D_{IMBA}. Finally, any additional delay that is introduced by the designer in order to assure that all data bits have traversed the network will be called the synchronization delay and represented as D_{SYN}. D_{SYN} may be

different for the crossbar and banyan networks.

In the next section relationships for these delays as functions of the various network parameters are derived.

2.2.1. INCREMENTAL CROSSBAR NETWORK

For the incremental crossbar network the average delay can be determined quite easily from the model shown in Figure 4. Notice that since this represents one plane (or slice) there will be $\lceil B'/B \rceil$ of these planes. Assume that each switch module represents a crossbar of N*N complexity and each is implemented on a single chip. The pin drivers for the output pins for each module are also located on the chip. It is clear that for this arrangement the number of modules in an average path is $\lceil N'/N \rceil$ and each intermodule path has the same delay characteristic D_{IMCB}. Therefore the average delay D'_{CB} through such a network is given by the expression:

$$D'_{CB} = \left\lceil \frac{N'}{N} \right\rceil D_{CB} + \left\lceil \frac{N'}{N} \right\rceil D_{IMCB} + D_{SYNCB} \qquad [3]$$

Note that a circuit switched design is assumed here with no pipelining present from module to module.

2.2.2 BANYAN NETWORK

The number of switch modules in the average path through a banyan network is $\log_N N'$ and the number of intermodule paths is also $\log_N N'$. Here, because of the connection topology, the intermodule paths are not generally constant in length. The average delay, D'_{BA}, through such a network (assuming no delay penalty for blocking) is given by:

$$D'_{BA} = \left\lceil \log_N N' \right\rceil D_{CB} + \left\lceil \log_N N' \right\rceil D_{IMBA} + D_{SYNBA} \qquad [4]$$

2.3 PIN CONSTRAINTS

For a square N*N*B chip with N_k pins allocated to power, ground and control, the pin constraint for a switch module is given by:

$$N_p >= KBN + N_k \qquad [5]$$

where K = 4 for the incremental crossbar network and K = 2 for the banyan network. The equality is used in subsequent computations since it is advantageous to use as many available pins as possible. Two cases may be considered. Case 1 will consist of the situation where the number of input and output pins is much larger than N_k (i.e., KBN >> N_k) and thus we can approximate N from equation 5 as:

$$N = \left\lfloor N_p/KB \right\rfloor \qquad [6]$$

This is typical of a clocked system where a small number of clock lines are needed to synchronize all the data lines.

Case 2 encompasses the situation where N_k is not neglible and there is a control line overhead associated with the data paths. Here we make the assumption that the number of control lines is proportional to the number of ports, N, on an individual chip. That is, N_k = QN where Q is a constant. Then N can be expressed as:

$$N = \left\lfloor N_p/(KB+Q) \right\rfloor \qquad [7]$$

This would be the appropriate model, for example, if network chips communicated with each other in an asynchronous manner and the control line overhead consisted of request/acknowledge pairs (Q = 2).

3. CHIP COUNT MINIMIZATION

As a first approximation consider large networks, large datapath widths and chips with a large number of pins. Then the ceiling functions can be removed from [1] and [2] and the floor functions from [6] and [7]. N_{cb} and N_{ba} are now approximated to be continuous functions as shown below:

$$N_{cb} = \frac{B'(N'**2)}{B(N**2)} \qquad [8]$$

$$N_{ba} = \frac{B'N'}{BN} \log_N N' \qquad [9]$$

Assume that the maximum number of pins that are available is used and consider Case 1 where N is given by equation 6. Substituting equation 6 ($K = 4$) in equation 8, and equation 6 ($K = 2$) in equation 9 we have:

$$N_{cb1} = \frac{16BB'(N'**2)}{N_p**2} = K_{cb}B \qquad [10]$$

$$N_{bal} = \frac{2B'N'\log N'}{N_p(\log N_p - \log 2B)} = \frac{K_{ba}}{\log N_p - \log 2B} \qquad [11]$$

For a given pin constraint N_p, and overall network requirements N' and B', K_{cb} and K_{ba} are constants. Minimizing N_{cb1} and N_{bal} for this case requires that B be minimized. The smallest datapath width possible is $B = 1$, hence with this model N should be selected to be N_p/K. This is a reassuring result since it corresponds to experience with memory chip design where the slice width is generally taken as one bit. Note however, that this was obtained with a continuous approximation to equations 1 and 2. The result is that while $B = 1$ yields a minimum number of chips in most cases, there are situations where other values of B are better. For instance, with a banyan network with $N_p = 60$, N' = 128 and B' = 16, a $B = 1$ solution yields $N_{bal} = 160$, while a $B = 2$ solution yields $N_{bal} = 144$.

For case 2 where N_k is not negligible equation 7 is used for N and substituted back in [8] and [9] to give:

$$N_{cb2} = \frac{(4B+Q)^2 B'N'^2}{BN_p^2} = \frac{K_{cb}(4B+Q)^2}{16B} \qquad [12]$$

$$N_{ba2} = \frac{K_{ba}(2B+Q)}{2B(\log N_p - \log(2B+Q))} \qquad [13]$$

The derivatives of N_{cb2} and N_{ba2} with respect to B can now be taken, and the values of B and N which minimize the chip count obtained.

For the case of T' an incremental crossbar, the number of chips N_{cb2} is minimized when $B = Q/4$. Thus for a request/acknowledge pair associated with each chip datapath ($Q = 2$), B would be selected as 1. While this is true for almost all cases considered, the continuous model approximation should be checked when N' is less than 64 or B' is less than 16 (e.g., For N' = 32 N_p = 75, Q = 2 and B' = 16; B = 1 yields N_{cb2} = 144: B = 2 yields N_{cb2} = 128).

For the case of T' a banyan network, an equation can be derived for obtaining the optimum B and N. These results indicate that the continuous model does not yield optimum or near optimum values in many situations, and that a search procedure working directly with equation 2 must be utilized. For instance, for N_p = 90, N' = 512, Q = 2 and B' = 16, an optimum B = 4 yields N_{ba2} = 684. Note that using B = 1 in this case results in N_{ba2} = 1152. This is not unusual, and in most cases (N_p < 140) where Q $\geq$ 2, a choice of B = 1 will be nonoptimal.

The discrete relations for the crossbar and banyan chip counts (equations 1 and 2) were solved using optimal values of N and B, and the chip count was

obtained as a function of the parameters N_p, N' and network type T'. Sample

results are presented in Figures 6 and 7. Figure 6 illustrates how the total

number of chips varies as a function of the network size assuming that B' = 16

and that the pin constraint is fixed at N_p = 90. The banyan requires fewer

chips than the crossbar implementation and this agrees with the observation

that the crossbar grows as 0(N'**2) while the banyan grows as 0(N'LogN').

The change due to altering Q from 0 to 2 is also depicted and, as would be

expected, increasing Q requires a larger number of chips for both the banyan

and the crossbar. Figure 7 presents the case in which Q is fixed at zero

and the pin constraint is a variable. Again the banyan requires fewer chips

than an equal size crossbar and fewer chips are needed in both networks

if the pins available per package is increased. Although not shown explicitly

in these graphs, the optimum value of B is 1 for the crossbar ($N_p \geq 64$),

while for the banyan the optimum B ranged from 1 to 4 ($N_p \geq 64$).

4.0 NETWORK DELAY MINIMIZATION

Next we determine expressions for each of the delays, D_{CB}, D_{IM}, and D_{SYN} and incorporate these into equations 3 and 4 to compute the average delay through the two networks. These expressions will then be examined to determine their minimum values.

4.1 CROSSBAR NETWORK

The value of D_{CB} has been developed by Franklin (10) using NMOS NOR gates for construction of the crossbar module and is given as:

$$D_{CB} = N[2.5mf\tau + \tau(1+2.25\alpha_{CB})] \qquad [14]$$

The parameters used in this relation are defined in Table 1 which also illustrates some typical values. The equation assumes a circuit switched crossbar, and uniformly distributed addressing of module output ports. The first term in the brackets represent the delay through an individual crosspoint on a module, while the second term is the delay between crosspoints on a module.

The delay encountered when a signal goes off the chip, propagates along an interconnecting path and enters another switch module is D_{IMCB}. A buffer (e.g., a series of inverters) must be included within the switch module to allow the minimum size transistor to drive the module pin and associated load with minimum delay. The buffer delay is determined by the gate capacitance of the minimum size transistor, the number of stages in the buffer, the capacitance of the pin being driven, the capacitance along the interconnecting path, and the capacitance of the receiving module pin. The buffer delay relation has been derived by Mead and Conway (14) who show that the minimum value occurs when exponentially sized cascaded inverters are used. The delay

in this case is given by:

$$D_{IMCB} = \tau e \, \log_e \beta \qquad [15]$$

where β is the ratio of the buffer load capacitance to the buffer input

transistor gate capacitance. The transistor gate capacitance, C_g, is merely

the capacitance per unit area times the gate area of the minimum transistor.

To determine the load capacitance we will assume that the driving and receiving

pin capacitances are equal and each has a value of C_{pin}. Further we postulate

that the modules for the crossbar network will be placed on a circuit board

and interconnected via printed circuit copper paths. Thus

$$\beta = (2C_{pin} + C_{path})/C_g \qquad [16]$$

Although the size of the individual switch modules varies as N, the spacing

between modules can be designed to be constant in the case of the crossbar

network and for most layouts will be less than one inch. For these cases the

pin capacitance dominates, hence:

$$\beta_{cb} = 2C_{pin}/C_g \qquad [17]$$

The synchronization delay is somewhat more difficult to quantify since

it depends upon the specific design technique used to determine that all bits

have traversed the network. Because of the parallel nature of communications

through such switching networks, the use of a global synchronous technique in

which the worst case delay through the network (e.g., top row, right column)

would be used to establish the clock period for all paths may be impractical.

Therefore many communication networks utilize some type of self-timing in

which timing information (e.g., data arrival) is imbedded in each message or

word. A reasonable design practice (that allows for additional uncertainty

in the path times due to temperature variations, changes in chip processing, etc.) is to include a tolerance or guard region that is proportional to the average delay time. As a consequence the average delay through the network can be expressed as:

$$D'_{CB} = (1+K_s)\left\lceil\frac{N'}{N}\right\rceil(D_{CB}+D_{IMCB}) \qquad [18]$$

or

$$D'_{CB} = K_1\left\lceil\frac{N'}{N}\right\rceil(D_{CB}+D_{IMCB}) \qquad [19]$$

where $K_1 = 1 + K_s$. If we now substitute equations 14, 15 and 17 into equation 19 we have:

$$D'_{CB} = K_1\left\lceil\frac{N'}{N}\right\rceil N[2.5mf\tau + \tau(1+2.25\alpha_{CB}) + (\tau e \log_e\beta_{CB})/N] \qquad [20]$$

Our computer simulation results have shown that for the crossbar network with zero control lines (e.g., $Q = 0$) the continous form of equation 20 usually gives the same results as the discrete form. Therefore we shall replace $\left\lceil\frac{N'}{N}\right\rceil N$ by N'. Finally, if we use the pin limitation constraint specified by equation 7 with $K = 4$ we have:

$$D'_{CB} = K_1N'[2.5mf\tau + \tau(1+2.25\alpha_{CB}) + (\tau(4B+Q)e \log_e\beta_{CB})/N_p] \qquad [21]$$

The following terms can be treated as constants in this analysis:

$$2.5mf\tau + \tau(1+2.25\alpha_{CB}) = A_0$$

$$[22]$$

$$\tau e \log_e\beta_{CB} = A_1$$

so we can express D'_{CB} as

$$D'_{CB} = K_1N'[A_0 + \frac{A_1}{N_p}(4B+Q)] \qquad [23]$$

It is clear that the sum 4B + Q should be a minimum to achieve a minimum value of D'_{CB}. Since $B \geq 1$ this indicates that the minimum value of D'_{CB} should occur when B = 1 and Q = 0. Notice also that the average delay is directly proportional to N', and decreases to a minimum value as the number of pins N_p increases.

To obtain additional insight into the variation of network delay as a function of network parameters it is useful to consider the parameter values given in Table 1. Using these values we find that:

$$D'_{CB} = 1.1N'[5.6 + \frac{14.4}{N_p}(4B+Q)] \text{ nsec} \qquad [24]$$

Notice that if Q = 0 and B = 1 then for large N_p (i.e., $N_p \geq 64$) the second term in the brackets in equation 24 is small and the average delay can be approximated as:

$$D'_{CB} \approx 6.2N' \text{ nsec} \qquad [25]$$

4.2 BANYAN NETWORK

The average delay through the banyan network is given by equation 4. The value of D_{CB} is known from equation 14 and once again we assign a value to D_{SYN} that is proportional to the average path delay through the network. Thus the only remaining quantity to determine is the value for D_{IMBA}. In contrast to the crossbar, the separation between switch modules in the banyan is not constant, and thus C_{path} will vary according to the banyan level. Since the number of levels required for a specific configuration is not known a priori, the inclusion of a variable for C_{path} complicates the delay computation. However, under the assumption that the switch modules are pin limited rather

than area limited, essentially constant intermodule delay can be ensured, independent of the banyan level, by increasing the pin driver area as the level increases. This yields a constant value for D_{IMBA} which can be obtained as follows: Assume that the switch modules are placed on a printed circuit board which is square with a side S inches in length. Consider the maximum path length at the last level of the banyan, and assume that because of the complexity of this last level one half of the board is devoted to the interconnection paths. The maximum length of a path (using horizontal and vertical routing) will then be S/2 + S/2 = S inches. Since the capacitance of typical printed circuit paths is approximately 1pf/inch, the delay in driving this path is:

$$D_{IMBA} = \tau e \ \log_e((2C_{pin}+S)/C_g) \qquad [26]$$

As a consequence the average delay through the banyan network can be expressed as:

$$D'_{BA} = K_1 \lceil \log_N N' \rceil \ \{N[2.5mf\tau + \tau(1+2.25\alpha_{CB})] + \tau e \ \log_e((2C_{pin}+S)/C_g)\} \qquad [27]$$

Tests indicate that the continuous version of this equation is often a poor approximation to the discrete version, thus it is not considered here. Using the values from Table 1 the banyan delay can be expressed as:

$$D'_{BA} = 6.17 \lceil \log_N N' \rceil (N + 1.78) \qquad [28]$$

The discrete relations for the crossbar and the banyan delays (equations 20

and 27) were solved using optimal values of N and B, and the delays were obtained as a function of parameters N_p, N' and network type T'. Sample results are presented in Figures 8 and 9. Figure 8 illustrates how the average delay varies with network size assuming B' = 16 and N_p = 90. The banyan delay is consistently smaller than the crossbar for networks of reasonable size.

The increase in the delay through the crossbar is small when Q is altered from 0 to 2. The reason for this is clear from examining equation 24. Q is often small compared to 4B, and the entire term is divided by N_p thus often making it negligible when compared to the factor 5.6 in the expression. In the banyan case the delay does not increase at all over the range Q ϵ [0,2]. To understand this, consider the range of N and B. The minimum values of N and B are 1 while the maximum values may be obtained from the pin constraint expression [7]. Thus for example with N' = 512, B' = 16, Q = 0 and N_p = 90, N ϵ [1,45] and B ϵ [1,45] while for Q = 2, N ϵ [1,22] and B ϵ [1,44]. The value of N which minimizes the delay as given in [28] is well within these ranges (i.e., N = 5) and from expression 7, the resulting B for Q ϵ [0,2] and N_p ϵ [60,120] is also within its range. Thus changing Q and N_p will effect the value of B, however the values of the optimal N will remain constant, and thus the curves for delay will be the same over wide Q and N_p ranges. In effect the optimum N (and thus minimum delay) is not on the constraint boundary.

5. CHIP COUNT-TIME PRODUCT MINIMIZATION

The chip count-time product P, can be obtained by multiplying the appropriate equations given earlier in the paper. This involves the product of equations 1 and 3 for the crossbar, and 2 and 4 for the banyan. Earlier discussion indicated that for reasonable size networks, both chip count and delay were minimized in the crossbar case with $B = 1$. Consequently the product is also minimized with this choice (for $N' \geq 64$, $B' \geq 16$).

For the banyan, the situation is more complex and a computer search for the optimum B and N values must be undertaken. Consider for example the case of $Q = 0$ and $N' = 512$. Table 2 shows the values of N,B which optimize the number of chips, the delay, and chip count-time product. It's clear from the table that the B and N values required for minimizing the chip count-time product fall between those needed for minimization of the chip count and delay measures by themselves. The count minimization is achieved by attempting to place as large a network as possible on a given chip. Delay minimization is achieved by balancing the delays associated with the module network, and the delays associated with increasing the number of levels in the overall network. From equation 28 one sees that placing as large a network on a chip as possible is not the best strategy from a delay point of view. Note that this analysis does not consider delays associated with network blocking. These can have a significant effect in a saturated network. Introducing these delays will have the effect of requiring larger modules since these modules are nonblocking crossbar networks, and hence will lower delays due to blocking.

Values for N and B which minimized the chip count-delay product P

were obtained for both network types over a range of N', N_p and Q values. Sample results of P versus N', as a function N_p, Q and T' are given in Figures 10 and 11. As expected P increases with increasing N' and increasing Q, and decreases with increasing N_p. Once again the banyan does better than the crossbar on this overall performance measure.

6. SUMMARY AND CONCLUSIONS

This paper concerned the design of multiple processor interconnection networks. Such networks can be characterized as having N' inputs and N' outputs, each B' bits wide. Construction of large networks requires partitioning of the $N'*N'*B'$ network into a collection of $N*N$ switch modules of data size B (B < B') each implemented on a single chip and interconnecting them with a specific interchip network type, T'. The major constraint in the VLSI environment is the pin limitation, N_p, of the individual modules; these are allocated between data and control lines, Q. This paper presented a methodology for selecting the optimum values for N and B given values of the parameters, N', B', T', Q, and N_p. Models for both the banyan and crossbar networks (T') were developed and arrangements yielding minimum: a) number of chips (e.g., switch modules), b) average delay through the network (e.g., maximum bandwidth), and c) product of number of chips and delay, were presented. The results show that for the crossbar a bit slice approach (B = 1) produces the optimum arrangement, while for the banyan the optimum is achieved with multiple bits per module. The impact of the number of control lines on chip count, delay and product were also discussed.

The analysis presented made a number of assumptions whose effects are

being further investigated. In particular the role of blocking in the banyan case, and the potential gain which would accrue from a pipelined design are under investigation. In addition, the problem of synchronization between network planes is being studied.

Parameter	Symbol	Units	Typical value
minimum feature size	λ_{min}	um	3
minimum gate area	$A_{min} = 4\lambda^2$	$(um)^2$	36
gate capacitance	C_g	pf	$1.4*10^{-2}$
switch module pin capacitance	C_{pin}	pf	5
transit time	τ	nsec	0.5
NOR gate logic levels per crosspoint	m	—	2
NOR gate fanout	f	—	2
metalization path capacitance to transistor gate capacitance ratio (within switch module)	α_{CB}	—	0.1
guard region multiplier	K_s	—	0.1
printed circuit path capacitance	C_{path}	pf	1pf/inch
length of side of printed circuit board	S	inches	12

TABLE 1:   TIME DELAY PARAMETERS

	N	B	CHIP COUNT	CHIP DELAY (nsec)	COUNT–DELAY PRODUCT ($*10^3$)
$N_p = 60$					
COUNT MINIMIZATION	30	1	576	392	226
DELAY MINIMIZATION	5	6	1236	168	207
PRODUCT MINIMIZATION	10	3	936	218	204
$N_p = 90$					
COUNT MINIZIZATION	45	1	384	578	222
DELAY MINIMIZATION	5	8	824	168	138
PRODUCT MINIMIZATION	11	4	564	237	133
$N_p = 120$					
COUNT MINIMIZATION	60	1	288	763	220
DELAY MINIMIZATION	5	11	824	168	138
PRODUCT MINIMIZATION	10	6	468	218	102

TABLE 2:   BANYAN NETWORK MINIMIZATION RESULTS
(N' = 512, Q = 0, B' = 16)
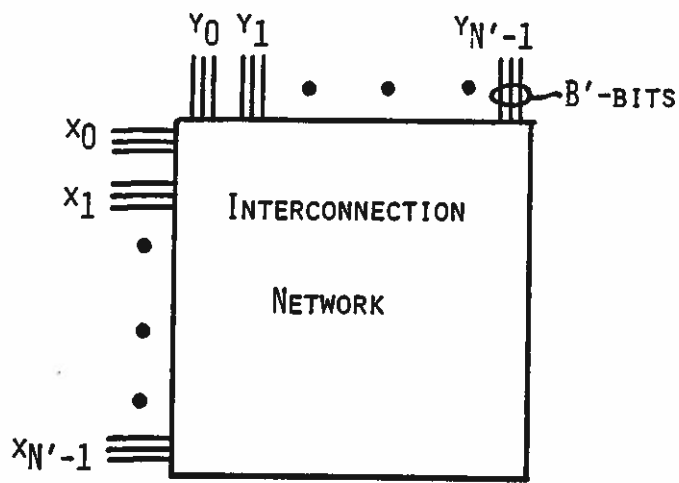
FIGURE 1: AN N'*N' NETWORK
(B' WIDE DATAPATHS)

BIT SLICE 0

BIT SLICE 1

BIT SLICE 2

BIT SLICE 3
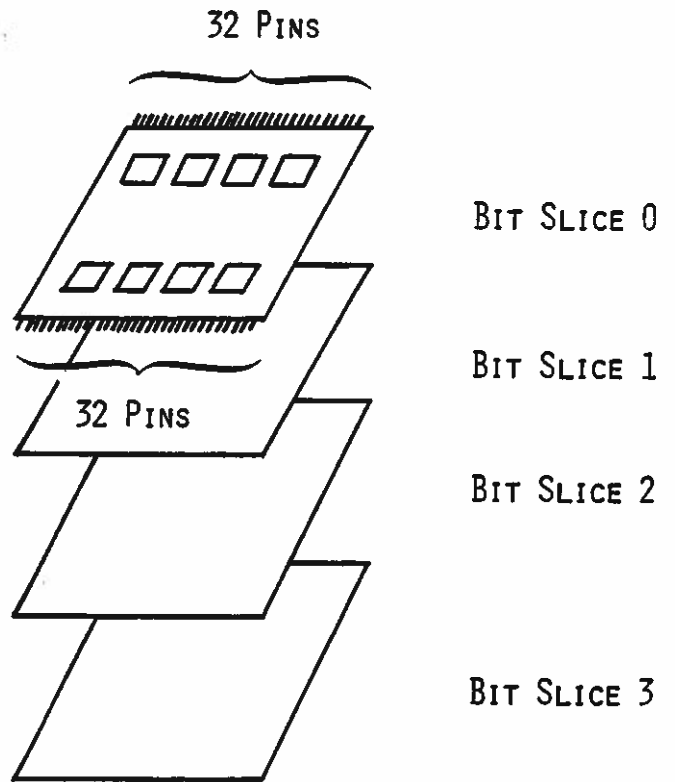
FIGURE 3: THE FOUR PLANES USED TO IMPLEMENT A
16*16*8 NETWORK. FIGURE 2 SHOWS A
SINGLE PLANE.
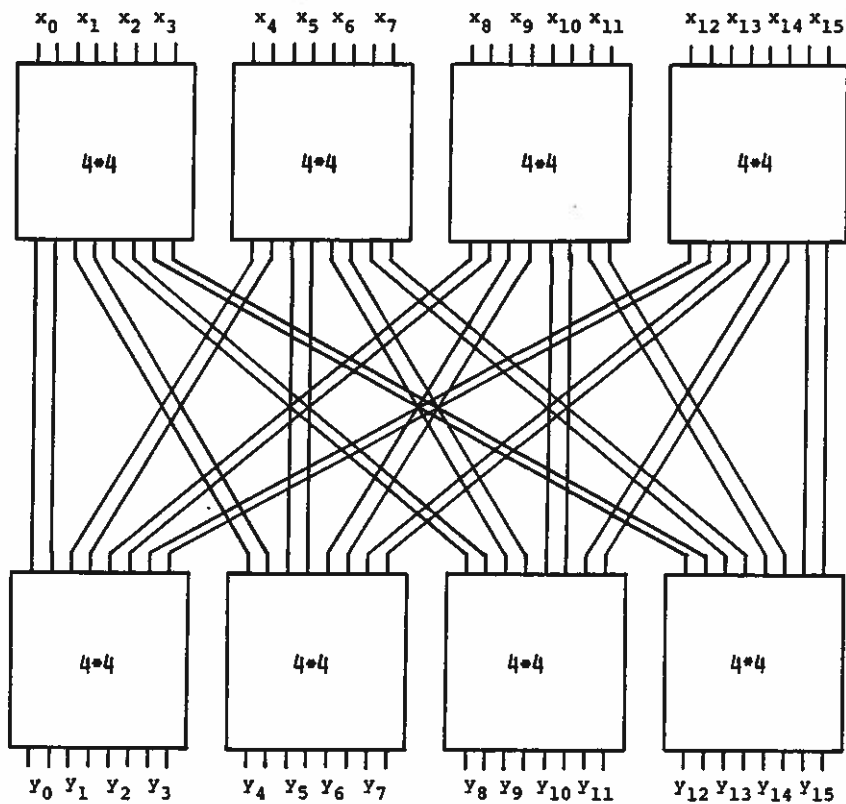
FIGURE 2: DECOMPOSITION OF A 16*16*8 NETWORK USING 4*4*2
NETWORK CHIPS (ONE PLANE OF FOUR SHOWN. CONTROL
AND POWER PINS NOT SHOWN).

INPUTS

OUTPUTS

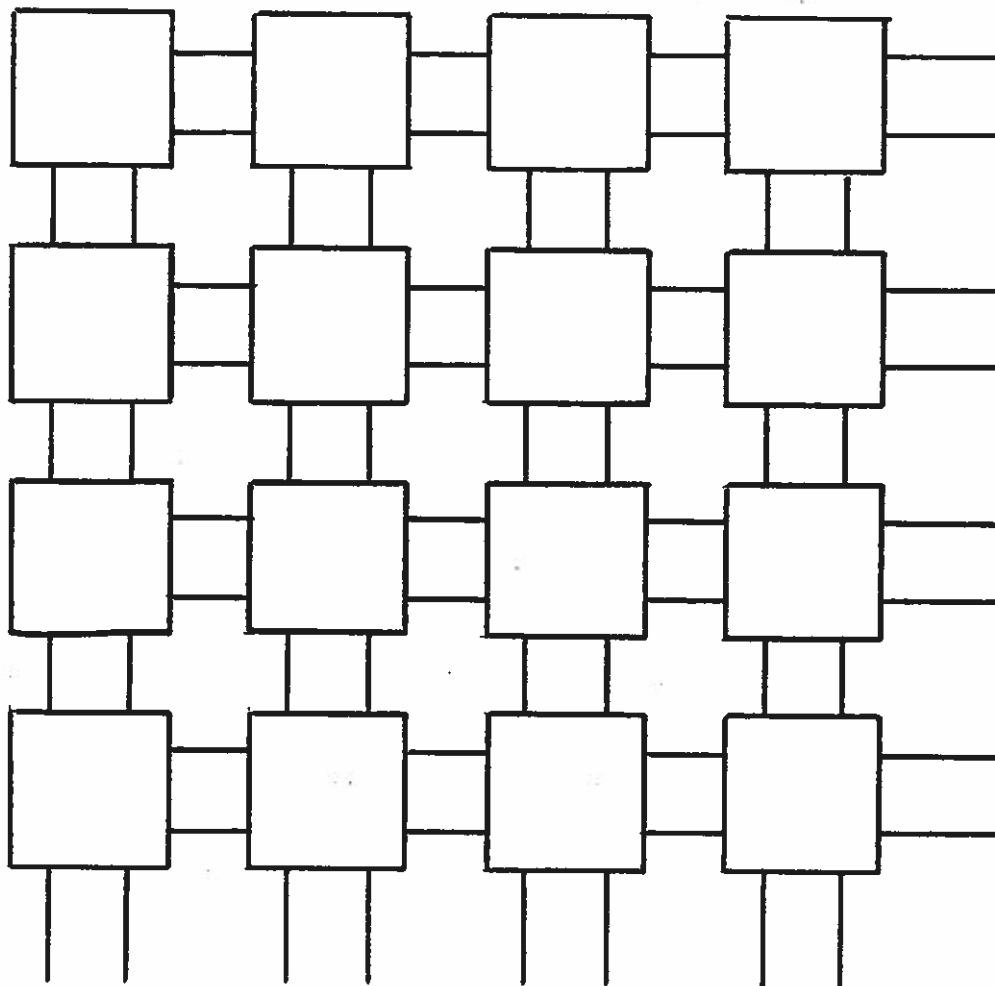FIGURE 4: AN 8*8*1 NETWORK COMPOSED OF 2*2*1 CHIP COMPONENTS
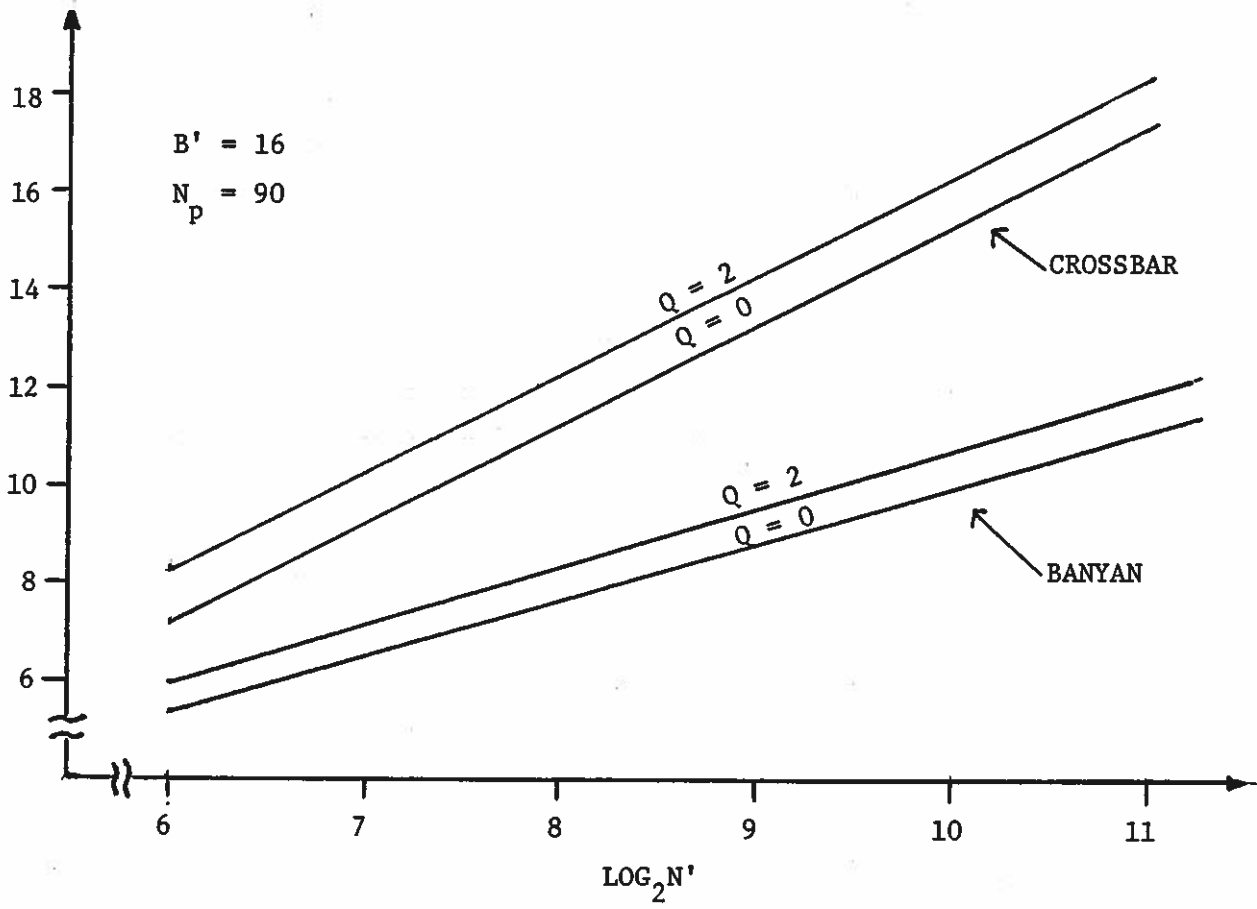ARRANGED IN AN INCREMENTAL CROSSBAR CONFIGURATION

INPUTS

OUTPUTS

FIGURE 5: AN 8*8*1 NETWORK COMPOSED OF
2*2*1 CHIP COMPONENTS ARRANGED
IN A BANYAN CONFIGURATION

FIGURE 6: NUMBER OF CHIPS, N_{Chips}, VERSUS NETWORK SIZE N'.
(VARYING T' AND Q)
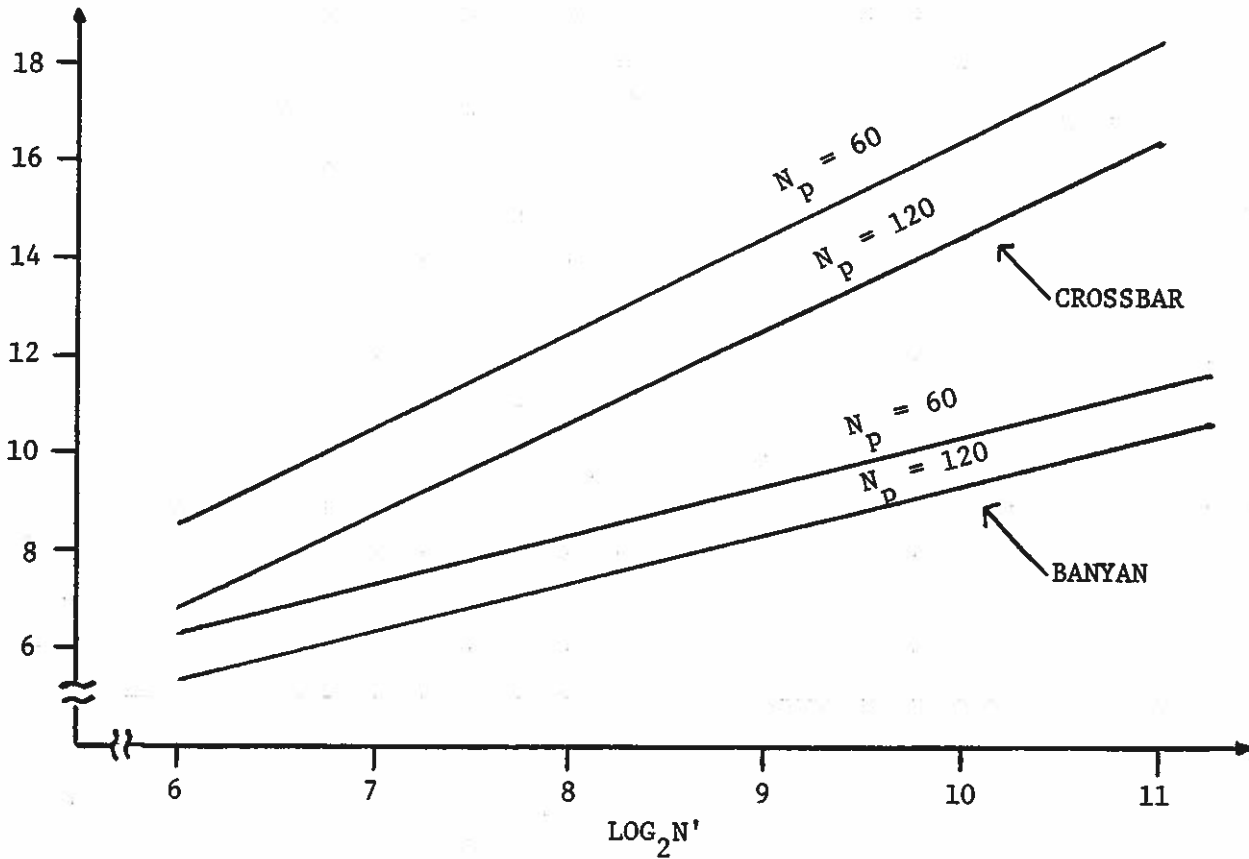
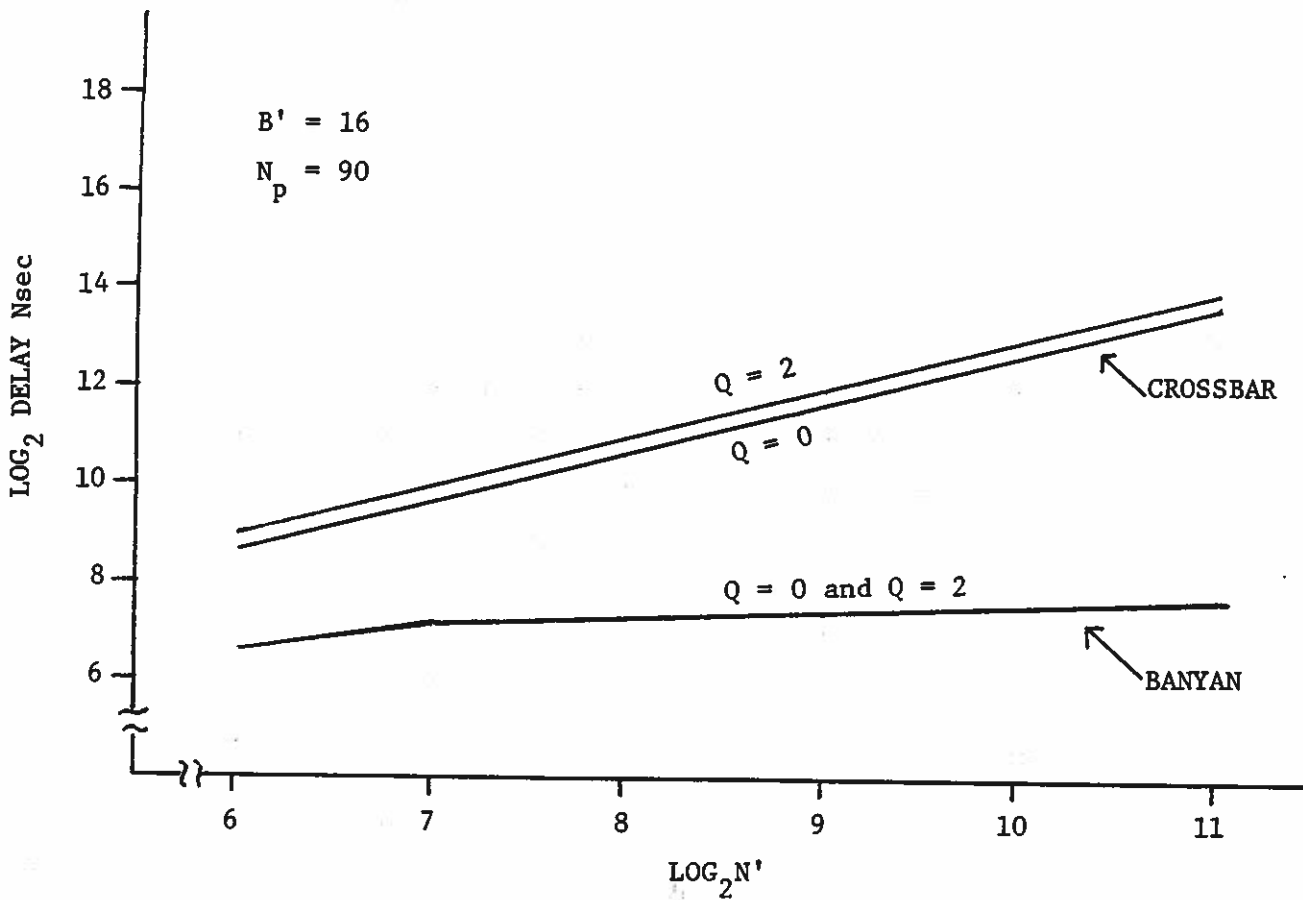FIGURE 7: NUMBER OF CHIPS, N_{Chips}, VERSUS NETWORK SIZE N'

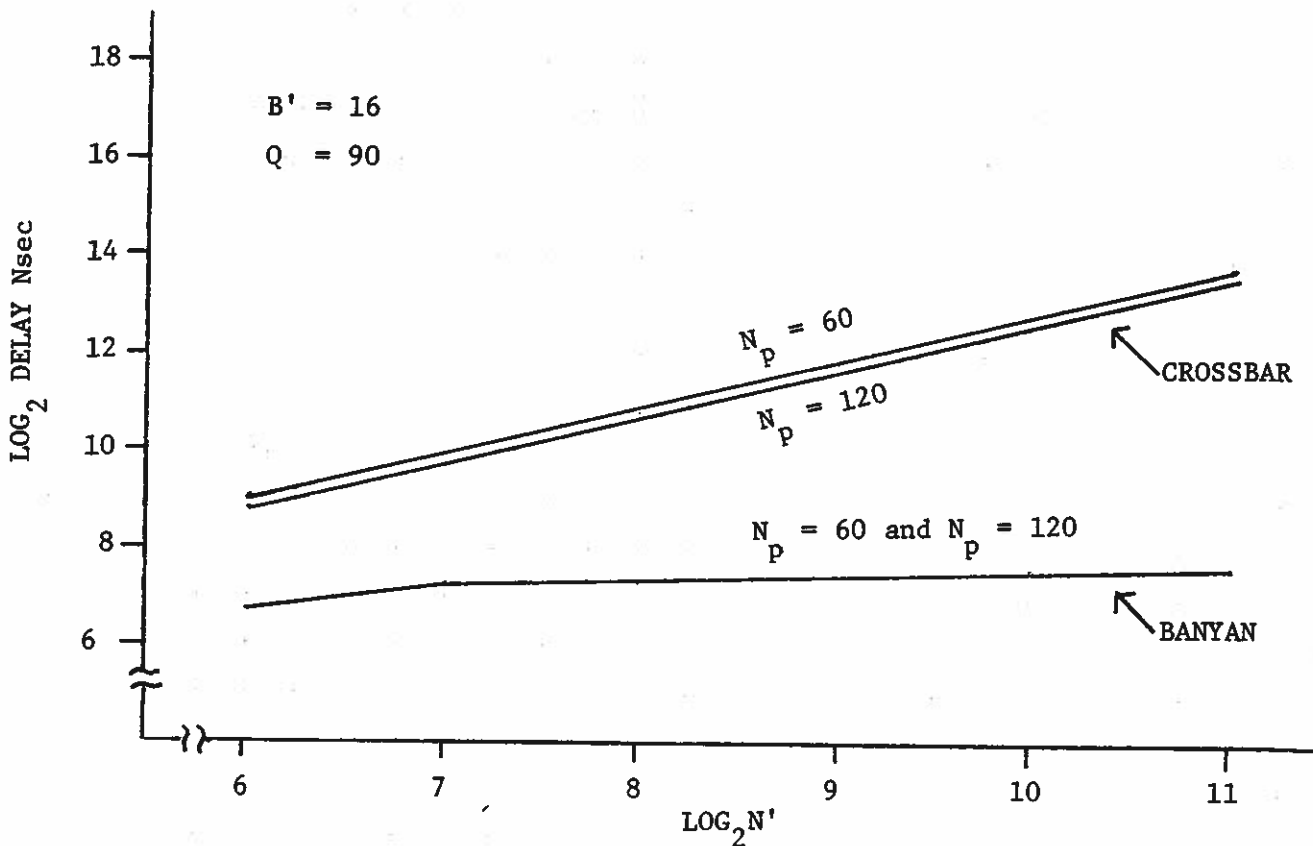FIGURE 8: DELAY VERSUS NETWORK SIZE, N'
(VARYING T' and Q)

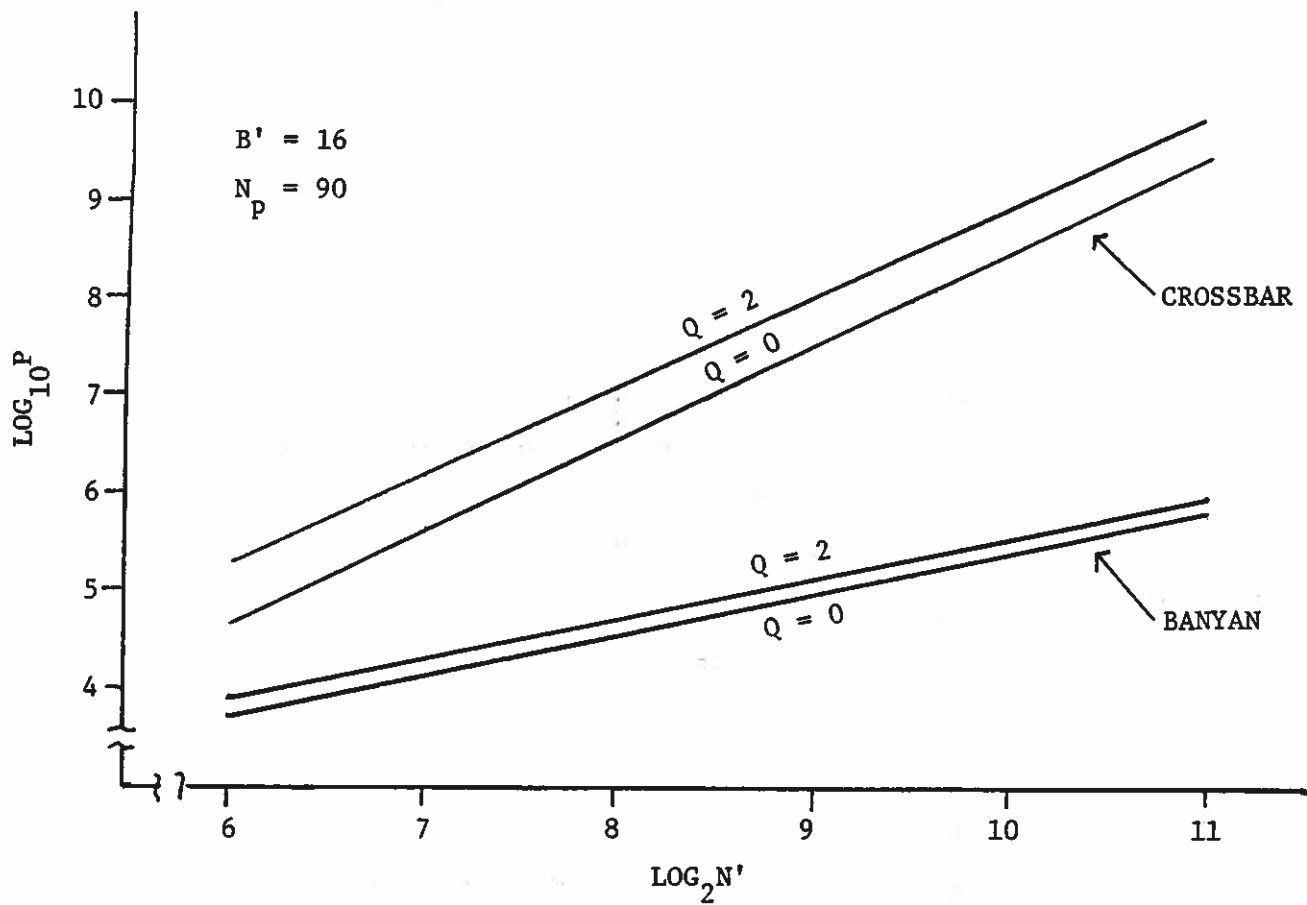FIGURE 9: DELAY VERSUS NETWORK SIZE, N'
(VARYING T' AND N_)

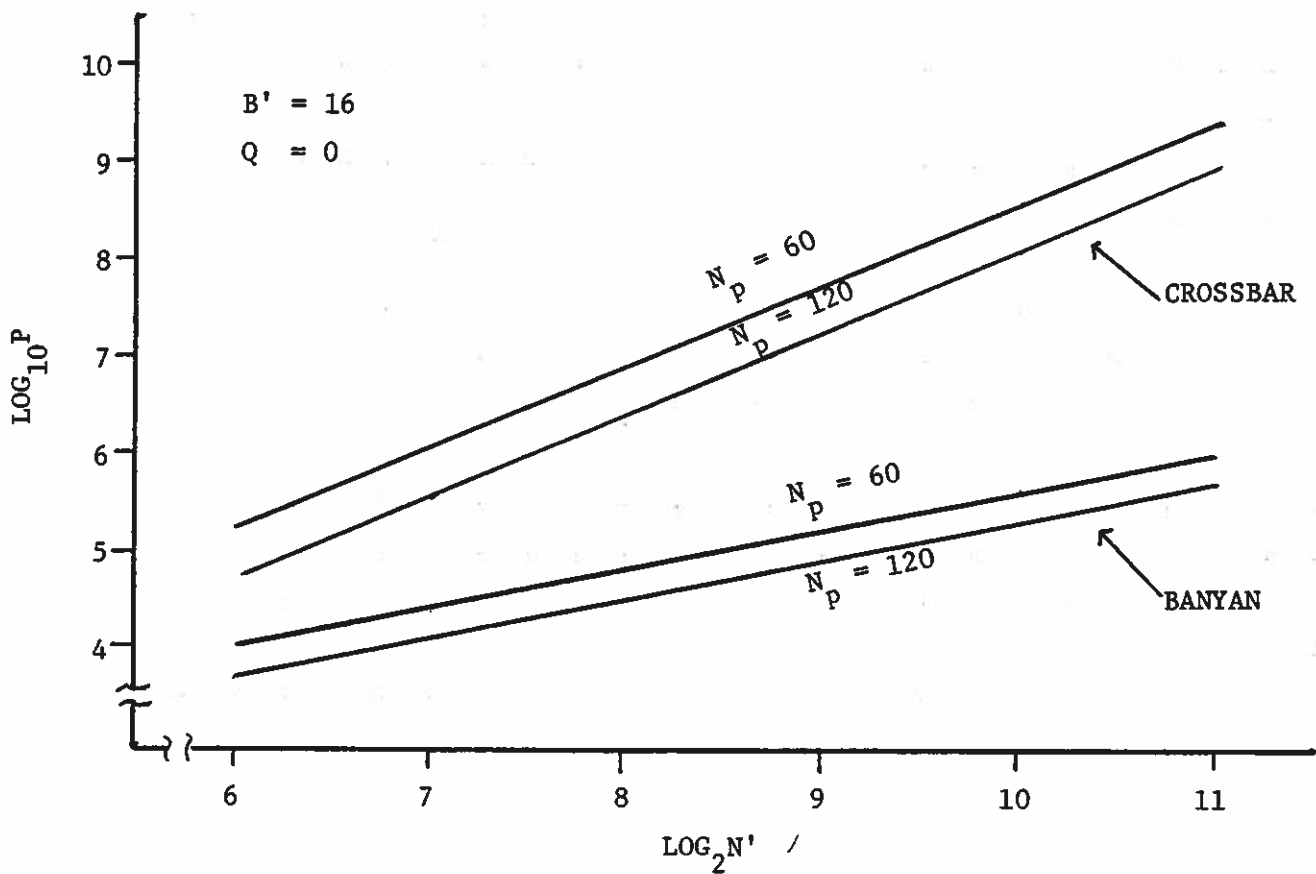FIGURE 10:   PERFORMANCE MEASURE, P, VERSUS NETWORK SIZE N'
(VARYING T' AND Q)

FIGURE 11:   PERFORMANCE MEASURE, P, VERSUS NETWORK SIZE N'
(VARYING T' AND N_p)

References

1. Swan, R.J., et.al., "Cm*-A Modular Multi-Microprocessor", Proc. NCC (1977).

2. Dennis, J.B., and Misunas, D.P., "A preliminary architecture for a basic data-flow processor", Proc. 2nd Ann. Symp. on Comp. Arch. (Dec. 1974).

3. Sejnowski, M.C., et. al., "An overview of the Texas Reconfigurable Computer", Proc. AFIPS Nat. Comp. Conf. (1980).

4. Sullivan, H. and Baskow, T.R., "A Large Scale, Homogeneous, Fully Distributed Parallel Machine I", Proc. 4th Ann. Symp. on Computer Architecture (March 1977).

5. Benes, Y.E., Mathematical Theory of Connecting Networks and Telephone Traffic, Academic Press, New York (1965).

6. Siegel, H.J.; McMillen, R.J., and Mueller, P.T., Jr., "A Survey of interconnection methods for reconfigurable parallel processing systems", Proc. 1979 Nat. Comp. Conf. (June 1979).

7. Anderson, G.A., and Jensen, E.D., "Computer interconnection structures: taxonomy, characteristics, and examples", ACM Comp. Sur. 7,4 (Dec. 1975).

8. Thurber, K.J., "Interconnection networks - a survey and assessment", Nat. Comp. Conf. (May 1974).

9. Franklin, M.A.; Kahn, S.A., and Stucki, M.J., "Design Issues in the Development of a Modular Multiprocessor Communications Network", Proc. of the Annual Symposium on Comp. Architecture (April 1979).

10. Franklin, M.A., "VLSI Performance Comparison of Banyan and Crossbar Communications Networks", IEEE Trans. on Comp. C-30,4 (April 1981).

11. Goke, L.R. and Lipoviski, G.J., "Banyan Networks for Partitioning Multiprocessor Systems", The First Ann. Symp. on Comp. Arch., University of Florida, Gainesville, Florida (1973).

12. Lawrie, D.H., "Access and Alignment of Data in an Array Processor", IEEE Trans. on Comp., Vol. C-24, No. 12 (Dec. 1975).

13. Pease, M.C., "The indirect binary n-cube microprocessor array", IEEE Trans. Comp. C-26,5 (May 1977).

14. Mead, C. and Conway, L., Introduction to VLSI Systems, Addison-Wesley Pub. Co. Reading, MA (1980).