New Jersey Institute of Technology
Digital Commons @ NJIT

Theses

# Investigation of reduced hypercube (RH) networks : embedding and routing capabilities 

Michalis A. Sideras<br>New Jersey Institute of Technology

Follow this and additional works at: https://digitalcommons.njit.edu/theses
Part of the Electrical and Electronics Commons

## Recommended Citation

Sideras, Michalis A., "Investigation of reduced hypercube (RH) networks : embedding and routing capabilities" (1994). Theses. 1697.
https://digitalcommons.njit.edu/theses/1697

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Theses by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.

## Copyright Warning \& Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If $a$, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page \# to: last page \#" on the print dialog screen

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

# ABSTRACT <br> Investigation of Reduced Hypercube (RH) Networks: Embedding and Routing Capabilities 

by<br>Michalis A. Sideras

The choice of a topology for the intercomection of resources in a distributedmemory parallel computing system is a major design decision. The direct binary hypercube has been widely used for this purpose due to its low diameter and its ability to efficiently emulate other important structures. The aforementioned strong properties of the hypercube come at the cost of high VLSI complexity due to the increase in the number of communication ports and channels per node with an increase in the total number of nodes. The reduced hypercube ( $R H$ ) topology, which is obtained by a uniform reduction in the number of links for each hypercube node. yields lower complexity interconnection networks compared to hypercubes with the same number of nodes, thus permitting the construction of larger parallel systems. Furthermore, it has been shown that the $R H$ at a lower cost achieves performance comparable to that of a regular hypercube with the same number of nodes. A very important issue for the viability of the RH is to investigate the efficiency of embedding frequently used topologies into it. This thesis proposes embedding algorithms for three very important topologies, namely the ring, the torus and the binary tree. The performance of the proposed algorithms is analyzed and compared to that of equivalent embedding algorithms for the regular hypercube. It is shown that these topologies are emulated efficiently on the RH. Additionally, two already proposed routing algorithms for the $R H$ are evaluated through simulation result.s.

by<br>Michalis A. Sideras

# A Thesis <br> Submitted to the Faculty of New Jersey Institute of Technology in Partial Fulfillment of the Requirements for the Degree of Master of Science in Electrical Engineering 

Department of Electrical and Computer Engineering


## APPROVAL PAGE

Investigation of Reduced Hypercube (RH) Networks: Embedding and Routing Capabilities

Michalis A. Sideras

Dr. Sotirios Q. Ziavras, Thesis Advisor
Assistant Professor of Electrical and Computer Engineering, NJIT Date

Dr. Denis Blackmore, Committee Member Date
Professor of Mathematics, NJIT

Or. John D. Carpineflí, Committee Member Date
Associate Professor of Electrical and Computer Engineering, NJIT

## BIOGRAPHICAL SKETCH

Author: Michalis A. Sideras
Degree: Master of Science in Electrical Engineering
Date: January 1994

## Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering, New Jersey Institute of Technology, Newark, NJ, 1994
- Bachelor of Science in Computer Engineering, New Jersey Institute of Technology, Newark, NJ. 1992

Major: Electrical Engineering

## Presentations and Publications:

Sotirios G. Ziavras and Michalis A. Sideras, "Facilitating High-Performance Image Analysis on Reduced Hypercube ( $R H$ ) Parallel Computers," $3^{r d}$ International Workshop on Parallel Image Analysis, Submitted for Publication.

This work is dedicated to
my parents
Antonis and Penelope

## ACKNOWLEDGMENT

The author wishes to express his sincere gratitude to his advisor, Professor Sotirios G. Ziavras, for his guidance, friendship, and moral support throughout this research.

Special thanks to Professors Denis Blackmore and John D. Carpinelli for serving as members of the committee and for providing valuable comments and suggestions for this thesis.

The author greatly acknowledges the asssistance of professors Gregory Kriegsmann and Michael Porter of the Department of Mathematics in the form of a graduate assistanship.

An additional word of thanks to Professor Michael Porter for being so supportive and understanding to the author as his supervisor in the computer lab of the Department of Mathematics and for making his high-tech computer equipment available to him from which this thesis has greatly benefited.

## TABLE OF CONTENTS

Chapter Page
1 INTRODUCTION ..... 1
1.1 Importance of Parallel Computing Systems ..... 1
1.2 Topologies ..... 3
2 HYPERCUBE VARIATIONS ..... 7
2.1 Existing Variations ..... 7
2.2 Reduced Hypercube Interconnection Network ..... 11
2.2.1 Emulating Hypercubes on Reduced Hypercubes ..... 15
3 ROUTING ALGORITHMS FOR REDUCED HYPER(UBES ..... 20
3.1 Routing Algorithm I ..... 20
3.2 Routing Algorithm II ..... 22
3.3 Simulation Results and Comparison ..... 23
4 MAPPING ALGORITHMS FOR REDUCED HYPERCUBES ..... 25
4.1 Mapping of a Ring onto an RH ..... 26
4.2 Mapping of a Torus onto an $R H$ ..... 34
4.2.1 Mapping of a $2-D$ Torus onto an $R H$ ..... 34
4.2.2 Mapping of a $\mu-D$ Torus onto an RH ..... 37
4.3 Mapping of a Binary Tree onto an RH ..... 39
5 CONCLUSIONS ..... 41
REFERENCES ..... 43

## LIST OF TABLES

Table
Page
3.1 Average distances of nodes in RH's according to Algorithms I and II ..... 24
4.1 The SOE's and SOX's for $n=2$ ..... 28
4.2 Efficiency of torus mapping ..... 39

## LIST OF FIGURES

## Figure

## Page

1.1 Fundamental topologies (a) Ring (b) Torus (c) Binary tree ..... 4
2.1 Structure of the $C C C(4)$ ..... 8
2.2 A 2-cube and a 3-cube connected to form an incomplete hypercube ..... 10
2.3 Producing the $R H(2,1)$ from the 4 -cube by removing the red links ..... 13
2.4 The RH(3,1) ..... 14
2.5 Structure of the $R H(k, 2)$ ..... 16
2.6 The average dilation of edges for hypercube emulation on the $R H(k, n)$. as a function of $k$ and $n$ for $k+2^{n} \leq 26$ ..... 19
4.1 Mapping of a 32 -node ring onto the $R H(3,1)$ ..... 31
4.2 Mapping of the $4 \times 8$ torus onto the $R H(3,1)$ ..... 36
4.3 Mapping of the $2^{k-2} \times 60$ torus onto the $R H(k, 2)$ ..... 38

## CHAPTER 1

## INTRODUCTION

### 1.1 Importance of Parallel Computing Systems

Historically the roots of supercomputers are traced in the military, intelligence and the scientific communities. During the period of the cold war supercomputers were indispensable in the design of nuclear weapons, radar tracking, and many other applications such as code breaking. All the above account for the very special treatment that the supercomputer industry received from the govermment.

The end of the cold war however did not bring the end of this industry. On the contrary, in the "economic war" which is taking shape now the role of supercomputers is and will be as important and even more important than it was during the period of the cold war. Supercomputers are growing into mainstrean business. industrial and design applications. For instance they are used in the design of antomobiles that will better protect passengers in crashes, for the design of internal combustion engines which burn fuel more efficiently, for the design of integrated circuits of unprecedented complexity and for pharmacenticals that are safer and more effective.

Originally efforts were heavily directed in the improvement of the raw speeds of the electronic components that make up a supercomputer. As a result of theso efforts there emerged the first class of supercomputers. Some representative examples include the CRAY-2, the IBM 3081/3084, as well as the Burroughs 1)-825. In spite of the fact that substantial, and in some cases impressive, improvements have been achieved, they were not enough to reach the continually increasing demands in speed. The speed of light is an unsurpassed limit to this approach. This has prompted computer engineers to seek different approaches.

The focus of both academia and industry has often shifted to a philosophy and approach which has been around for the last 15 years, that of massive paralletism. It is now accepted, astonishingly with a wide consensus, that massive paralletism is
the only way to achieve the target speedups and even to break the teraflop barrier ( 1 Teraflop $=1$ trillion floating point, operations per second).

This has led to the emergence of massively parallel processor (MPP) systems which are moving toward domination of the supercomputer market. MPP's are high performance architectures which are based on the interconnection of hundreds or thousands of microprocessor-based nodes as opposed to 4.8 or a few dozens of expensive, exotic processors in conventional supercomputers. The main advantage of MPP's is their support of scalability, that is the ability to design high performance architectures capable of accommodating thousands of processors. overcoming potential problems such as insufficient memory and communication overhead. The latter is a real issue for MPP's and it refers to the time it takes for the nodes to communicate in order to complete a task. It is obrious that this overhead is at the expense of the speedup gained by the parallelism of the task. Currently there are worldwide approximately an equal number of conventional supercomputers and powerful MPP's. There are about 500 of each kind. Both design techniques have achieved speeds approaching 10 gigaftops ( 1 gigaflop $=1$ billion floating point operations per second). As it has been alluded to carlier, the target of both industry and the government is to approach processing rates of a trillion floating-point operations per second ( 1 teraflop) sustained on actual applications. This is 100 times the processing power of today's best machines which run at about 10 gigaflops. The hope is that teraflop performance will be reached before the end of this decade. The target of teraflop performance is not arbitrary "There are certain useful tasks which need this kind of performance, such as for example the simulation with reasonably fine detail of the aerodynamics of an entire arcraft, the simulation of the global climate over a period of decades, and close to accurate weather forecasting.

### 1.2 Topologies

A distinct characteristic of an MPP computer is the way its processing elements (PE's) are interconnected, i.e. the topology of the intercomection network. Actually one of the ways to classify multiprocessors is to make distinctions based on the topology of their interconnection networks.

Some fundamental topologies for the development of parallel algorithms are the following:

- Ring (or linear array)
- Torus (or mesh)
- Binary Tree
- Pyramid

The first three topologies are shown in Fig 1.1.
For instance, the linear array is important for image coding, the mesh is indispensable for mumerical analysis and intermediate-level image processing, the binary tree is inherently the most suitable topology for divide-and-conquer tochniques, and the pyramid is suitable for multigrid operations and inage processing.

When it comes to the design and manufacturing of a commercial MPP, the objective is often to build a general purpose system. This is the case usually muless there is a specific demand and need for a specialized system. The term general purpose clearly implies and dictates that the topology of the computer should be able to efficiently emulate the fundamental structures like those listed above. Indeed, during the short history of this new industry, the most successful MPP's were designed with a general purpose topology for the interconnection of their PE's.

One very important topology which has been and still is central to research efforts in this area is the direct binary $n$-cube, otherwise called the $n$-dimensional

(a)

(b)

(c)

Figure 1.1 Fundamental topologies (a) Ring (b) Torus (c) Binary tree
hypercube. The direct binary $n$-cube is a special case of the direct $k$-ary $n$-cube with $n$ dimensions and $k$ nodes in each dimension [11]. The terms $n$-dimensional hypercube and $n$-cube will be used interchangeably in the rest of this thesis to denote the direct binary $n$-cube. Simply described, an $n$-dimensional hypercube or $n$-cube consists of $2^{n}$ nodes. If unique consecutive binary $n$-bit addresses are assigned to its nodes, then a link (edge) exists between nodes whose addresses differ in a single bit. Consequently each node has $n$ links attached to it. The hypercube possesses some extremely important topological properties which make it a very good choice for the interconnection of PE's in MPP systems. The most important of these properties are the following:

1. Small diameter in large systems. The diameter of an interconnection network is defined as the maximum of the shortest distances between all pairs of nodes. For an $n$-cube, which has $2^{n}$ nodes, the diameter is cqual to $n$.
2. Small average internode distance in large systems which is defined as the average of the distances of all nodes from a reference node. It is equal to $\frac{n}{2}$ for the $n$-cube.
3. General purpose topology, since it can emulate other important structures very efficiently. Indeed, a considerable number of algorithms have been proposed for embedding several fundamental networks into the hypercube. Agorithms for the mapping of rectangular meshes have been proposed among others by Chan and Saad [13], and Johnsson [14]. Binary tree mappings were proposed among others by $W_{u}[15]$, Deshpande and Jevenin [3], Ho and Johnsson [6], and Johnsson [14]. Finally, algorithms for embedding pyramids have been designed among others by Chan and Saad [13], Lai and White [16], and Ziavras and Siddiqui [17].

The above advantages are a direct consequence of the high degree of connectivity of the nodes in the hypercube and its highly regular structure.

Several successful commercial products make use of the hypercube. The Thinking Machines CM-2 (Connection Machine-2), the NCUBE and the Intel iPSC are the most important. The CM-2 has up to 65,536 PE's which are simple 1 -bit processors. In contrast the other two machines have a relatively small number of powerful processors (up to 1,024 ). The relation between the number of processors and their complexity is not coincidental. The hypercube has a major disadvantage, namely its number of channels per processor increases as the $\log _{2}$ of $N$ where $N$ is the total number of processors. The resulting high VLSI complexity of the hypercube is a limiting factor on the feasible size of a system. This is exactly the reason why the CM-2 uses simple processors in order to have a large number of them.

The high VLSI complexity of the hypercube has forced several supercomputer manufacturers to seek different topologies for their new systoms. Nevertholess, as the hypercube remains a good choice due to its powerful properties, researchers are now directing their efforts on the development of hypercube-like topologies with lower VLSI complexities. One should expect from such a structure to maintain to a large extent the powerful properties of the hypercube but at the same tinc have a considerably lower VLSI complexity, thus permitting the construction of larger systems that can yield higher throughputs. Section 2.1 briefly describes some existing important hypercube-like topologies which are also called hypercube variations. 'This thesis investigates the routing and embedding capabilities of such a variation, namely the reduced hypercube ( $R H$ ) interconnection network. The reduced hypercube is a very promising hypercube variation which was proposed recently [2] and is described in detail in Section 2.2 .

## CHAPTER 2

## HYPERCUBE VARIATIONS

### 2.1 Existing Variations

In Chapter 1 the major drawback of the hypercube interconnection network was identified as its resistance to incremental growth due to high VLSI complexity for large systems. In order to mitigate this disadvantage but at the same time maintain to a high extent the very important properties of the hypercube, rescarchers have developed several variations of the regular hypercube. The extent to which the above mentioned objective has been achieved varies. The RH interconnection network. which is central to the research presented in this thesis, is a recent promising variation of the regular hypercube. A brief description of some important existing hypercube variations is deemed necessary and is in order.

The cube-connected cycles $C C C(n)$ interconnection network is presented first since it has a special relation with the $R H$. This relation will be explaned in Section 2.2. The $C C C(n)$ is made up of $2^{n} n$-node rings. Each entire ring is assigned an address just if it were a hypercube node in a $2^{n}$-node configuration. Furthemore. each individual ring node has a unique $\log _{2} n$ - bit address within the $n$-node ring. The overall address of each node in the $C C C(n)$ is obtained by the concatenation of the ring address with the node address within the ring. Nodes are connected in the following way. Each node connects to the node which has the samo lower part of its address and whose upper part of the address differs in the bit which is equal to the decimal representation of the lower part. In other words, each node implements a connection in the direction it represents. Each node with address i in a ring implements a connection in the $i^{t h}$ dimension of the $2^{n}$-node hypercube. The $C C C(n)$ has the important property that its connectivity factor, i.e. the number of edges per node, is equal to 3 , independently of the value of $n$. The $C C C(4)$ is shown in Figure 2.1.


Figure 2.1 Structure of the $C C C(4)$

The incomplete hypercube is another important variation of the hypercube. An incomplete hypercube is constructed by connecting two complete hypercubes of different sizes. Figure 2.2 shows the structure of an incomplete hypercube which is comprised of two maximal-sized complete hypercubes with 2 and 3 dimensions. respectively.

In spite of the fact that the incomplete hypercube atiempts to solve the incremental growth problem of the conventional hypercube, it has the major disadvantage that a considerable number of communication ports remain totally unused. For example, consider an incomplete hypercube with 2,560 nodes which is obtained by connecting a 2,048 -node hypercube and a 512 -node hypercube. Had these been two distinct hypercubes, they would have had 11 and 9 commmication ports per node, respectively. Since they have to be connected together to form the incomplete hypercube, they now have an additional port per node, thus bringing the number of ports per node to 12 and 10 , respectively. From the 2,048 nodes of the first hypercube only 512 of them will be comnected to the second hypercube. This implies that the rest of the nodes in the first hypercube (i.e. $2,048-512=1.536$ ) will have an unused port each, for a total of 1,536 unused communication ports. It is assumed that the same kind of processors are used for all nodes in cach hypercube. This is primarily dictated by practical design needs. In general, assuming that an no cube and an $n_{1}$-cube ( $n_{0}>n_{1}$ ) are connected to form an incomplete hypercube, the total number of unused communication ports is equal to $2^{n_{0}}-2^{n_{1}}$. This serves as an indication of the cost associated with unused resources. In spite of the fact that the incomplete hypercube structure seems to mitigate the increncontal growth problem for the hypercube, it does have, at least locally, a higher VLSI complexity. This is not desirable however because one of the original goals was to obtain hypercube variations that have lower overall VLSI complexity in order to make the construction of larger systems feasible.


Figure 2.2 A 2-cube and a 3-cube connected to form an incomplete hypercube

### 2.2 Reduced Hypercube Interconnection Network

Although [2] introduced reduced hypercubes ( $R H$ 's) which can be viewed as hierarchical structures with several levels, only the properties of structures with two levels were studied extensively. This thesis also focuses on the RH's with only two levels. RH's are obtained by uniformly removing at design time several edges from hypercubes with the same number of nodes.

A reduced hypercube $R H(k, n)$ contains a total of $N$ nodes where $N=2^{k+2^{n}}$. $k \geq n$ and $n>0$. For simplicity let the exponent term $k+2^{n}$ be denoted by $v$ as it will be used repeatedly. Each node of the $R H(k, n)$ is attached to $k+1$ bidirectional channels. In a regular hypercube with the same number $N=2^{\prime \prime}$ of nodes, each node is attached to $\nu$ bidirectional links. Therefore, each node in the N-node RHI has $\nu-(k+1)$ or $2^{n}-1$ links less than each node in the $N$-node regular hypercube

The $N$-node $R H(k, n)$ is constructed from the $N$-node regular hypercube by uniformly removing $2^{n}-1$ links from each of its nodes. To accomplish this, the $\nu$-bit addresses of hypercube nodes are first partitioned into two fields, whe $0^{t h}$ and $1^{\text {st }}$ fields, as follows. The $0^{\text {th }}$ field contains the $k$ least significant bits of the $\nu$-bit node address. This field represents the address of the node within a complete tecube. which will be referred to as a building block (BB). The $1^{\text {st }}$ field contains the $2^{n}$ most significant bits of the $\mu$-bit node address. It represents the address of the $B B$ that contains the node. In addition, a subfield is identified in the $0^{\text {th }}$ field, the $0^{\text {th }}$ subfield. It contains the $n$ most significant bits of the $k$-bit $0^{\text {th }}$ field. It represents the address of a $(k-n)$-dimensional subcube, which will be referred to as a subblock $(S B)$, within the $k$-cube $B B$ that contains the node.

To conclude, the address of each node in the $R H(k, n)$ is formed as shown in the following diagran, where the symbol " $e$ " denotes concatenation and it will be used as such throughout this document.

In order to reduce the $\nu$-cube into the $R H(k, n)$, out of the $\nu$ bidirectional links of each hypercube node the following two sets are kept, leaving $k+1$ links to each node.

Set 1: The $k$ links of the $l$-cube that traverse the $k$ lowest dimensions (i.e., dimensions 0 through $k-1$ ) and connect the referenced node with $k$ distinct nodes are kept. As a result, a complete $k$-dimensional building block ( $B B$ ) that includes the referenced node is maintained.

Set 2: This set contains only one link which is also present in the original $\nu$-cube. This link is the one which connects directly the referenced node with the node whose address differs only in the $m^{\text {in }}$ bit of the $1^{s t}$ field, where $m$ is the decimal value in the $0^{t h}$ subfield and $0 \leq m \leq 2^{n}-1$.

The resultant $R H(k, n)$ contains $2^{2^{n}} k$-cube $B B$ 's. A $B B$ address forms the $2^{n}$ most significant bits (i.e. the $1^{s t}$ field) of the $\nu$-bit addresses for contained nodes. Each $B B$ is divided into $2^{n}$ subblocks (SB's). Connections between pairs of $S B$ 's in different $B B$ 's are as follows: A node in a particular $S B$ of a particular $B B$ is connected to the node with the same $0^{\text {th }}$ field address which belongs to the $B B$ whose $2^{n}$-bit address differs only in the $m^{\text {th }}$ bit, where $m$ is the value in the $0^{t h}$ subfield of the former node.

Figure 2.3 shows the $R H(2,1)$ after the links in red are removed from the 4-dimensional hypercube. The $R H(3,1)$ is shown in Figure 2.4. The doted lines represent the inter-building block links. It was shown in [2] that the $R / H$ can mulate simultaneously, with dilation equal to one, several cube-connected cycles networks [8].


Figure 2.3 Producing the $R H(2,1)$ from the 4 -cube by removing the red links


Figure 2.4 The RH(3.1)

Example 2.1 Figure 2.4 shows the structure of the RH(3,1). Note that there are $2^{n}=2^{1}=2 S B^{\prime}$ s per $B B$. Each $B B$ is a complete $3-$ cube, since $k=3 . B B$ addresses appear above each $B B$. In this case they consist of 2 bits. $S B$ addresses appear inside $S B$ boxes in the upper left corner. For this example, the $S B$ addresses are 1-bit quantities. To understand better the way nodes in the $R H$ are interconnected, a few nodes are examined. Periods are used to separate from left to right the $B B$ address $\left(1^{\text {st }}\right.$ field $)$, the $S B$ address, $\left(0^{\text {th }}\right.$ subfield), and the node address in the subblock ( $k-n$ bits). The node which belongs to $B B=00, S B=1$ and has addres.s 11 in the $S B$ has an address denoted by 00.1.11. This node is connected with a set I link to the node with the same lower field in the $B B \overline{0} 0=10$, because the $0^{\text {th }}$ subfield contains the subscript of the bit to be complemented. Therefore. the node 00.1.11 connects to the node 10.1.11. Similarly 01.1.11 connects to 11.1.11.

A more general case is shown in Figure 2.5 [2]. It is the structure of the $R H(h, 2)$ where the large squares represent the $k$-cube building blocks. The mumbers above the squares represent in decimal the $B B$ addresses and the numbers within the quadrants of large squares are the $S B$ addresses in decimal. For simplicity, the actual nodes within the squares are not shown. Each curved line represents $2^{k-2}$ bidirectional communication channels; this is also the number of $P E$ 's in each $S B$. It is implied that each $P E$ in a $S B$ is connected to the $P E$ with the same $0^{\text {th }}$ field address in the $S B$ where the curved line leads.

### 2.2.1 Emulating Hypercubes on Reduced Hypercubes

Given the fact that an $R H$ is equivalent to a regular hypercube with fewer links, as explained previously, it is evident that the performance of the topology may sometimes degrade to some extent. The largest degradation appears for algorithms designed explicitly for the regular hypercube. It is desired to find the degree of this degradation not only for the purpose of theoretical analysis but also for a more


Figure 2.5 Structure of the $R H(k .2)$
practical reason. There is a plethora of important algorithms which have already been developed for the regular hypercube, therefore how well the $R H$ cmulates the regular hypercube is important. Indeed, this was investigated in [2] and the most important findings are presented below.

The dilation of edges associated with the chosen hypercube mapping must be found for evaluation of the performance. The dilation measures the increase in the communication overhead when compared to one-hop data transfers in the regular hypercube. Let the regular $\nu$-dimensional hypercube and the target $R H(k, n)$ contain the same number of nodes; that is $2^{\nu}$ where $\nu=k+2^{n}$. Assume that nodes from the regular hypercube are mapped to nodes of the $R H$ with the same address. The following theorem [2] presents the resultant dilations of edges.

Theorem 2.1 For the emulation of the $\left(k+2^{n}\right)$-dimensional hypercube on the reduced hypercube $R H(k, n)$ with the same number of nodes, the dilations of the edges incident to a single node of the hypercube are: ! for $k+1$ of them and $2 p+1$ for $\binom{n}{p}$ of them, where $p=1,2, \ldots, n$ and, $\binom{n}{p}$ represents the number of distinct p-combinations of $n$ items.

Example 2.2 The dilations of the edges incident to a single node of the RH(5,2) for the emulation of the 9-dimensional hypercube are 1,3 and 5 for 6,2 and 1 edge. respectively. Similarly, the dilations of the edges incident to a single node for the emulation of the 16 -dimensional hypercube on the $R H(8,3)$ are $1,3,5$ and 7 for 9 , 3, 3 and 1 edge, respectively.

There are two other important metrics: the maximum and average dilations of edges for hypercube emulation on the $R H$. The following two corollarics provide the means for their calculation [2].

Corollary 2.1 The maximum dilation of edges for hypercube emulation on the $R H(k, n)$ is equal to $2 n+1$.

Corollary 2.2 The average dilation of edges for hypercube emulation on the $R H(k, n)$ is equal to

$$
\frac{k+(n+1) 2^{n}}{k+2^{n}}
$$

The average dilation of edges for the last two examples is 1.88 and 2.5 , respectively. Figure 2.6 [2] shows the average dilation of edges for hypercube emulation as a function of $k$ and $n$, with $1 \leq n \leq 4,2 \leq k \leq 10$ and $k+2^{n} \leq 26$. Therefore, the emulation of hypercubes with up to 26 dimensions is considered in Figure 2.6. The numbers next to graph points represent the numbers of dimensions in the emulated hypercubes. It must be mentioned that the curves for $n=1,2$ and 3 represent realistic cases with respect to the actual implementation of the respective target systems with the current technology due to the relatively low complexity of RH's. Figure 2.6 shows that the average dilation of edges for hypercube emulation is relatively small. This observation guarantees small performance degradation for the implementation of hypercube algorithms on RH's. The effect of dilation is reduced significantly from left to right for the set of four well known packet switching techniques: store-and-forward, virtual cut-through, circuit switching, and wormhole routing.


Figure 2.6 The average dilation of edges for hypercube emulation on the $R H(k, n)$, as a function of $k$ and $n$ for $k+2^{n} \leq 26$

## CHAPTER 3

## ROUTING ALGORITHMS FOR REDUCED HYPERCUBES

Two distributed routing algorithms for RH's which were proposed in [2] are discussed in this chapter and their comparative analysis follows based on simulation results. Both algorithms consist of three steps and their first two steps are identical. The E-cube routing algorithm for regular hypercubes uses dimension ordering so that messages traverse in a predetermined increasing or decreasing order the dimensions where the source and destination addresses differ. This ordering is required to avoid the creation of deadlocks during the transmission of multiple messages. The increasing dimension order is used throughout this thesis. The description of the routing algorithms below ignores potential deadlock problems for the transmission of multiple messages. Virtual chamels can be introduced to make both routing algorithms deadlock-free [2].

### 3.1 Routing Algorithm 1

The presentation of the first routing algorithm for the $R H(k, n)$ is very detailed for the purpose of clarity. The execution steps in this routing algorithm are as follows [2].

Step 1. Apply the E-cube routing algorithm for regular hypercubes to the least significant $k-n$ bits of the source address, if the source and destination addresses differ in any of these bits; use dimension ordering, starting with dimension 0 and following the increasing order. This step implements lecube routing within the source $S B$.

Step 2. If the destination was reached, then stop. Otherwise, let $\lambda$ be the subscript of the most significant bit position where the current and destination addresses differ (it can be found by an $X O R$ operation). If $\lambda<k$, then apply the $E$-cube
routing algorithm to the $0^{t h}$ subfield of the current address and stop when the destination PE is reached. Otherwise, let $m$ be the value represented by the $0^{\text {th }}$ subfield of the current address. If the current and destination addresses differ in the $m^{t h}$ bit of their $1^{s t}$ field, then transmit the message to the PE whose address differs from the current address only in the bit with subscript $k+m$ (a direct comection exists for the implementation of this fransfer) and go back to Step 2. Otherwise, if the two addresses do not differ in the $m^{\text {th }}$ bit of their $1^{\text {st }}$ field, then go to Step 3.

Step 3. Find the bit in the $1^{\text {st }}$ fied that differs in the current and destimation addresses and whose subscript in this field differs in the smallest number of bits from the $n$-bit value in the $0^{t h}$ subfield of the current address. For multiple subscripts corresponding to the smallest number of bits, choose one of them at random. To find the neighbor that then receives the message, cary out $i$-cube routing within the k-cube building block to the nearest PE that can correct the aforementioned bit in the $1^{\text {st }}$ field. Go to Step 2.

Example 3.1 Assume the RH $(7,3)$. If the source and destination addresses are source address $=00010010.010 .0101$ and destination address $=01110110.101 .1101$ then according to Routing Algorithm I the path is given by the following sequence of addresses.
00010010.010 .0101
00010010.010 .1101
00010110.010 .1101
00010110.110 .1101
01010110.110 .1101
01010110.111 .1101
$01010110.1 \underline{0} 1.1101$
01110110.101 .1101

The underlined bit was changed in the corresponding step.

### 3.2 Routing Algorithm II

This algorithm performs better than the first routing algorithm for source and destination addresses that differ in a large number of bits in their $1^{\text {st }}$ field. This routing algorithm uses the binary reflected Gray Code.

Definition 3.1 The (cyclic) binary reflected Gray code of $n$ bits $R G C(n)$ is defined recursively by $R G C(n)=\left\{0 \bullet R G C(n-1), 1 \bullet R G C^{-1}(n-1)\right\}$, where "* denotes concatenation, $R G_{C} C^{-1}(n-1)$ denotes the sequence derived by reversing the order of elements in the sequence $R G C(n-1)$, and $R G C(1)=\{0,1\}$.

Example 3.2 $R G C(2)=\{00,01,11,10\}$ and $R G C(3)=\{000,001,011,010,110$. $111,101,100\}$.

Any Gray code has the property that any two neighboring codes in the sequence of all possible $2^{n} n$-bit numbers differ in a single bit. The actic version of the RCC' ( $n$ ) is used throughout this thesis. The first two steps of the new algorithm are identical to those of routing algorithm l. Its third step follows.

Step 3. Create a sequence of $n$-bit numbers that starts with the value in the $0^{\text {th }}$ subfield of the current node, contains the offsets of bits that differ in the $1^{\text {st }}$ field of the current and destination addresses, and ends with the value in the $0^{t h}$ subfield of the destination node. Order the sequence of offsets as follows. First, create two candidate sequences that contain all of the offsets in the order they appear in the $R C C(n)$, when forward or backward traversal is carried out, respectively. From the two sequences choose the one with the smaller total number of 1 bits in the XOR results between consecutive pairs of offsets,
including the current and destination subfields. This choice results in a smaller number of changes for the $0^{\text {th }}$ subfield and thus reduces the total number of intermediate nodes in the path. Then apply the E-cube routing algorithm to the $0^{t h}$ subfield of the current address in order to go closer to the node whose lower subfield is the same as the next offset in the chosen sequence. Go to Step 2.

### 3.3 Simulation Results and Comparison

Besides the diameter, one very important metric for any interconnection network is its average internode distance. It should be made clear that the average distance in the $R H(k, n)$ depends on the actual routing algorithm which is used. The average distance of the $n$-dimensional hypercube is given by $\frac{n}{2}$. Table 3.1 shows the average distances in the $R H(n, n)$, with $n=1,2,3$ and 4 , for the first routing algorithm and a variation of the second algorithm. More specifically strict dimension ordering with the reflected Gray code is assumed for the second algonthm (the forward and backward traversal of the Gray code produce the same result when all possible sourcodestination pairs are considered [2]). The simulation for the case $n=4$ is very time consuming when all possible source-destination pairs are considered. Therefore, our simulation assumes all possible destinations for the source node with address 0 , and the table contains the average for the forward (FWD) and backward (BWD) traversal of the Gray code. The fourth column is what is obtained if at each iteration the shortest distance according to both algorithms is chosen. The last column contains the average distance when for each source-destination pair the minimum of the distances is chosen with forward or backward traversal of the reflected Gray code. It comes as no surprise that the fourth column holds the shortest distances. It was observed that algorithm II performs better than algorithm I for pairs of source and destination addresses that differ in a large number of bits in their $1^{s t}$ field.

Table 3.1 Average distances of nodes in RH's according to Algorithms I and II

| n | Alg. I | Alg II (FWD or BWD) | Best of I\&II | Best of Alg. II |
| :---: | :---: | :---: | :---: | :---: |
| 1 | 2.000000 | 2.000000 | 2.000000 | 2.000000 |
| 2 | 4.746094 | 4.875000 | 4.621094 | 4.625000 |
| 3 | 9.851364 | 10.492188 | 9.559372 | 9.929688 |
| 4 | 19.387329 | 21.196258 | 19.148174 | 20.548462 |

## CHAPTER 4

## MAPPING ALGORITHMS FOR REDUCED HYPERCUBES

The most desired algorithm for the mapping of a source topology onto a target topology is one which yields an optimal mapping. The following definition is pertinent.

Definition 4.1 A mapping is optimal if the source topology is a proper subgraph of the target topology. In this case there is a one-to-one mapping of source nodes to target nodes and the resultant dilations of all source edges are 1 . The dilation of a source edge is defined as the length of the shortest path that connects the images of its two incident nodes in the target topology.

In this chapter, algorithms are presented for the mapping of three very important structures onto the RH topology. More specifically, mapping algorithms are proposed for the ring (or linear array), the $\mu-\mathrm{D}$ ) torus (or mesh), and the binary tree structures. Each mapping is evaluated in terms of the dilation of edges and the utilization of resources. They are also compared to existing mappings of the same structures onto the regular hypercube.

Before the formal introduction of the mapping algorithms, a defintion pertaining to them is in order.

Definition 4.2 The cyclic binary $U$ Gray code of $n$ bits $\operatorname{VGC(n)}$ is defined recursively by $U G C(n)=\left\{R G C(n-1) \bullet 0, R G C^{-1}(n-1) \bullet 1\right\}$, and $U G C(1)=R G C(1)=$ $\{0,1\}$.

Example $4.1 U G C(2)=\{00,10,11,01\}$ and $U G C(3)=\{000,010,110,100,101$. $111,011,001\}$.

### 4.1 Mapping of a Ring onto an RH

In the case of the regular hypercube the mapping of the ring onto it is optimal and the algorithm is straightforward. It is based on the property of the $R C C$ that any two successive codes in the $R G C$ differ in a single bit. Since nodes in the hypercube whose addresses differ in a single bit are directly connected, a ring may be obtained by assigning in ascending order hypercube addresses to ring nodes using the RGC sequence.

The RH has considerably fewer links per node compared to the regular hypercube. As explained in Section 2.2, nodes in the RH whose addresses differ in a single bit are not necessarily linked together as they are in the regular hypercube. In fact, two nodes in the $R H(k, n)$ are linked together if their addresses differ in any single bit in the $0^{\text {th }}$ field, or if they differ in only that bit of the $1^{\text {st }}$ field whose offset in this field is equal to the decimal value in their $0^{\text {th }}$ subfield. Therefore. the $R$ © 6 alone cannot be used for optimal mapping. The definitions of building block ( $B B$ ) and subblock $(S B)$ presented earlier in Section 2.2 are essential for understanding the proposed mapping algorithm hercafter.

For the $N$-node $R H(k, n)$, the algorithm generates a sequence containing all possible $N$ addresses ensuring that successive addresses represent nodes of the $R H$ which are linked together directly. The sequence of addresses is generated by traversing the $R H(k, n) B B$ by $B B$ according to the $R C C\left(2^{n}\right)$. All nodes in a $13 B$ are traversed consecutively, therefore each $B B$ is visited only once. The first node with which the traversal of a particular $B B$ starts will be referred to as the Node of Entry (NOE) into that BB. Furthermore, the $S B$ that contains this NOE will be referred to as the Subblock of Entry (SOE). Similarly, the last node traversed in a $B B$ and its corresponding subblock will be referred to as the Node of Exit (NOX) and Subblock of Erit (SOX), respectively. The following theorem is the basis of the proposed algorithm.

Theorem 4.1 If $B B$ 's in the $R H(k, n)$ are traversed according to the $R G C\left(2^{n}\right)$, each $B B$ is entered from $S B 0$ and exited from $S B \delta$, or vice versa, where the $S B$ address $\delta$ takes values from 1 to $2^{n}-1$. This will be denoted by $0 \longleftrightarrow \delta$.

Proof. Every other code in the $R G C\left(2^{n}\right)$ differs from its succeeding code only in the least significant bit. Therefore, the SOX for half of the $B B S$ is $S B O S B$ 0 is then the $S O E$ for the other half of $B B$ 's because only $S B$ 's with the same address are connected together in $R H$ 's.

Given the SOE and SOX of each $B B$ in the $R H$, the way the nodes of a particular $S B$ are traversed is determined by the algorithm below for the case $0 \longrightarrow \delta$, i.e. $S O E=0$ and $S O X=\delta$. Let the ordered sequence $T_{0-0}=\left\{T_{0}, T_{1}, \ldots, r_{2^{k}-1}\right\}$ contain the node addresses of the $B B$ in the order in which they are traversed so that the NOE belongs to $S B 0$ and the NOX belongs to $S B \delta$.

The following algorithm is valid for $k>n$, whereas the case $k=n$ is investigated at the end of this section.
STEP 1 Choose $\tau_{0}=\overbrace{00 \ldots 0}^{k \text { bies }}$

STEP 2 Start following the $k$-bit $R G C(k)$. If the parity (i.e. the number of l's) of the SOX address is even stop traversing the $R G C(k)$ at $S=S_{k-1} S_{k-2} \ldots S_{0}=$ $\delta_{n-1} \delta_{n-2} \ldots \delta_{0} \bullet 0 \bullet \overbrace{000 \ldots 0}^{k-n-1 \text { bits }}$, where $S$ is a $k$-bit address and $\delta_{n-1} \delta_{n-2} \ldots$ $\delta_{0}$ is the $n$-bit representation of $\delta$. Otherwise, for odd parity stop at $\delta_{n-1} \delta_{n-2}$ $\ldots \delta_{0} \bullet 1 \overbrace{000 \ldots 0}^{k-n-1 \text { bits }}$.

STEP 3 The traversal of the remaining nodes is the next task.

If $k-n>1$, follow the $k$-bit $U G C(k)$. Otherwise, for $k-n=1$ :

1. If the parity of the $S O X$ address is even, follow the $U G C(k)$.
2. If the parity of the $S O X$ address is odd, follow the $U G C^{-1}(h)$.

Table 4.1 The SOE's and SOX's for $n=2$

| $B B$ Address | SOE | SOX | Form |
| :---: | :---: | :---: | :---: |
| 0000 | 11 | 00 | $3 \rightarrow 0$ |
| 0001 | 00 | 01 | $0 \rightarrow 1$ |
| 0011 | 01 | 00 | $1 \rightarrow 0$ |
| 0010 | 00 | 10 | $0 \rightarrow 2$ |
| 0110 | 10 | 00 | $2 \rightarrow 0$ |
| 0111 | 00 | 01 | $0 \rightarrow 1$ |
| 0101 | 01 | 00 | $1 \rightarrow 0$ |
| 0100 | 00 | 11 | $0 \rightarrow 3$ |
| 1100 | 11 | 00 | $3 \rightarrow 0$ |
| 1101 | 00 | 01 | $0 \rightarrow 1$ |
| 1111 | 01 | 00 | $1 \rightarrow 0$ |
| 1110 | 00 | 10 | $0 \rightarrow 2$ |
| 1010 | 10 | 00 | $2 \rightarrow 0$ |
| 1011 | 00 | 01 | $0 \rightarrow 1$ |
| 1001 | 01 | 00 | $1 \rightarrow 0$ |
| 1000 | 00 | 11 | $0 \rightarrow 3$ |

Stop at $S^{\prime}=S_{k-1} S_{k-2} \ldots \bar{S}_{0}$
For the reverse case of $\delta \longrightarrow 0$, nodes are traversed in exactly the opposite order of that for the $0 \longrightarrow \delta$ case. The next step determines the SOE and SOX pair for each $B B$ in the $R H$. Given a $B B$ with address a. it.s. SOE has address equal to the offset of the bit where $\alpha$ differs from its immediately preceding code in the cyclic $R G C\left(2^{n}\right)$. Its $S O X$ has address equal to the offset of the bit where a differs from its immediately succeeding code in the cyclic $R C C\left(2^{n}\right)$. Table 4.1 illustrates the $S O E$ 's and $S O X$ 's for $n=2$.

Formally, assuming that the $2^{2^{n}} B B$ 's of the $R H(h, m)$ are visited using the $R G C\left(2^{n}\right)$, their $S O X$ 's are given by the exit sequence $E S(p)$ derived by $E S(p)=$ $\left\{E S^{\prime}(p), p\right\}$, where $E S^{\prime}(j)$ is found recursively by $E S^{\prime}(j)=\left\{E S^{\prime}(j-1), j, E S^{\prime}(j-1)\right\}$ and $E S^{\prime}(0)=0,1 \leq j \leq p$ and $p=2^{n}-1$.

Example 4.2 For the $R H(k, 1)$, we have $p=2^{1}-1=1$,
$E S^{\prime}(1)=\left\{E S^{\prime}(0), 1, E S^{\prime}(0)=\{0,1,0\}\right.$, and $E S(1)=\{0,1,0,1\}$.
Similarly, for the $R H(k, 2)$, we have $p=2^{2}-1=3, E S^{\prime}(3)=\left\{E S^{\prime}(2), 3, E S^{\prime}(2)\right\}$.
But $E S^{\prime}(2)=\left\{E S^{\prime}(1), 2, E S^{\prime}(1)\right\}=\{0,1,0,2,0,1,0\}$, therffore
$E S^{\prime}(3)=\{0,1,0,2,0,1,0,3,0,1,0,2,0,1,0\}$ and
$E S(3)=\{0,1,0,2,0,1,0,3,0,1,0,2,0,1,0,3\}$

From the $E S(p)$ sequence, the list of $S O E-S O X$ pairs of the form $0 \longleftrightarrow \delta$ can be derived. Let $E S(p)=\left\{\zeta_{0}, \zeta_{1}, \ldots, \zeta_{2^{2^{n}-1}}\right\}$. Then the sequence of these pairs
 the $S O X$ of any $B B$ is the $S O E$ of the succeeding $B B$ in the $R C C\left(2^{\circ}\right)$.

Example 4.3 For the RH( $k, 2$ ), as calculated in the previous example $S P=\{3 \rightarrow 0$. $0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 2,2 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 3,3 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 2$. $2 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 3\}$, as shown in Table 4.1.

The sequence $T_{S P[2]}$ that shows the order in which the nodes of the $i^{\text {th }} B B$ in the $R G C\left(2^{n}\right)$ are traversed is then produced, such that the first node belongs to the $S O E$ while the last node belongs to the $S O X$. Finally, each ring node is assigned an $R H$ address by concatenating a $B B$ address with the address of the node within the $B B$ as determined earlier. The following C-like program performs the concatenation.

```
for(i=0;i<2n;i++)
{
    for( }j=0;j<\mp@subsup{2}{}{k};j++
        {
            address[2k}\timesi+j]=RGC[i] T TSP[i][j]
        }
}
```

Theorem 4.2 Given an $N$-node $R H(k, n)$ with $k>n$, there is an optimal mapping of the $N$-node ring onto it.

Proof. The combination of the Gray codes $R G C$ and $U C C$ allows the traversal of all the nodes in every $B B$.

Example 4.4 Here is a complete example for the mapping of a ring onto the $R H(3,1)$.

1. The SOE/SOX pairs are of the form $0 \leftrightarrow 1$
2. $T_{0 \rightarrow 1}=\{000,001,011,010,110,100,101,111\}$ and
$T_{1 \rightarrow 0}=\{111,101,100,110,010,011,001,000\}$
3. $n=1 \Rightarrow p=2^{1}-1=1 \Rightarrow E S^{\prime}(1)=\{0,1,0\} \Rightarrow E S(1)=\{0,1,0,1\}$
4. $S P=\{1 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 1\}$
5. The ring node addresses are:
$00 \cdot T_{1-0}$
6. $T_{0-1}$
$11 \cdot T_{1-0}$
$10 \cdot T_{0-1}$
Expanding the T' scquences:
00111, 00101, 00100, 00110, 00010, 00011, 00001, 00000 .
01000, 01001, 01011, 01010, 01110, 01100, 01101, 01111,
$11111,11101,11100,11110,11010,11011,11001,11000$.
10000, 10001, 10011, 10010, 10110, 10100, 10101, 10111.
The mapping is shown in Figure 4.1. The links which are actually used for the mapping are represented by thick lines. Following the nodes in Figure 4.1 according to the above mapping sequence (along the thick links). It can be verified that the mapping is optimal since all nodes are used and the ditation of all edges is equal to 1 .


Figure 4.1 Mapping of a 32 -node ring onto the RH(3,1)

For $n=k$, which corresponds to the case where each $S B$ contains only one node, the mapping is not always optimal. The mapping algorithm presented for the case of $n>k$ is good for the case of $n=k$ except for the step in which the $T_{0-b}$ sequences are produced. Keep in mind that since each $S B$ contains only one node, the terms SOE and NOE and the terms SOX and NOX are equivalent, respectively. The terms SOE and SOX are used here. To obtain the $0 \rightarrow \delta$ sequences the following algorithm must be used.

Let $S O X=\delta=\delta_{n-1} \delta_{n-2} \delta_{n-3} \ldots \delta_{1} \delta_{0}$, where $\delta_{n-1} \delta_{n-2} \delta_{n-3} \ldots \delta_{1}=$ $R G C_{\phi}(n-1)$ and $\phi$ represents the index of the particular code in the $R C C^{\prime}(n-1)$ sequence.

STEP 1 Follow the $R G C(n)$.

- Stop at $R G C_{\phi-1}(n-1) \cdot \bar{\delta}_{0}$ if $R G C_{\phi}(n-1)$ has odd parity.
- Stop at $\delta_{n-1} \delta_{n-2} \delta_{n-3} \ldots \delta_{1} \bar{\delta}_{0}$ if $R C C_{o}(n-1)$ has cven parity:

STEP 2 Follow the $\|\left(G^{-1}(n)\right.$. Stop at $\delta$.

Example 4.5 Assume the RH $(2,2)$.

1. The SOE/SOX pairs are of the form $0 \leftrightarrow 1,0 \leftrightarrow 2,0 \leftrightarrow 3$
2. $T_{0-1}=\{00,10,11,01\}$ and $T_{1-0}=\{01,11,10,00\}$
$T_{0 \rightarrow 2}=\{00,01,11,10\}$ and $T_{2-0}=\{10,11,01,00\}$
$T_{0-3}=\{00,01,11\}$ and $T_{3-0}=\{11,01,00\}$
3. $n=2 \Rightarrow p=2^{2}-1=3 \Rightarrow E S^{\prime}(3)=\{0,1,0,2,0,1,0,3,0,1,0,2,0,1,0\} \Rightarrow$ $E S(3)=\{0,1,0,2,0,1,0,3,0,1,0,2,0,1,0,3\}$.

$$
\begin{aligned}
& \text { 4. } S P=\{3 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 2,2 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 3,3 \rightarrow 0, \\
& \quad 0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 2,2 \rightarrow 0,0 \rightarrow 1,1 \rightarrow 0,0 \rightarrow 3\}
\end{aligned}
$$

5. The ring node addresses are:
$0000 \cdot T_{3-0}$
0001 • $T_{0 \rightarrow 1}$
$0011 \cdot T_{1 \rightarrow 0}$
$0010 \cdot T_{0 \rightarrow 2}$
$0110 \cdot T_{2-0}$
$0111 \cdot T_{0 \rightarrow 1}$
0101 • $T_{1 \rightarrow 0}$
$0100 \cdot T_{0 \rightarrow 3}$
$1100 \cdot T_{3 \rightarrow 0}$
$1101 \cdot T_{0 \rightarrow 1}$
$1111 \cdot T_{1-0}$
$1110 \cdot T_{0-2}$
$1010 \cdot T_{2-0}$
$1011 \cdot T_{0-1}$
$1001 \cdot T_{1 \rightarrow 0}$
$1000 \cdot T_{0 \rightarrow 3}$

It can be seen in the previous example that one node is not used in each of the cases $T_{0-3}$ and $T_{3-0}$. The following theorem is pertinent.

Theorem 4.3 For $n=k$. $B B$ 's where the address $\delta$ of the $S O E$ or the $S O X$ is one of the following two types:

- the most significant bit is 0 and the parity of the remaining bits is even
- the most significant bit is 1 and the parity of the remaining bits is odd one $S B$ will not be utilized in the mapping of the ring. $S B$ 's of the form just described will be referred to as special $S B$ 's.

Proof. Assume the mapping of $S B^{\prime}$ 's onto a mesh of size $2^{n-1} \times 2$. The row numbers of the mesh are encoded using the $R C C(n-1)$ while the column numbers are 0 and 1. Each point in the mesh then represents the $S B$ whose address is obtained by concatenating the encoded row and column numbers. The mapping atgorithm traverses SB's in a serpentine-like manner in Step 1, while in Step 2 it traverses $S B$ 's using straght lines. For special $S B$ 's mentioned in the theorem, the $S B$ which is in the same row whth the $S O X$ is omitted in the traversal.

Corollary 4.1 The mapping of a ring onto the RH(k, 1) with the same number of nodes is optimal.

Proof. Since $n=1$, according to Theorem 4.3 there are no special $S B$ s.

If a dilation of two can be tolerated, then all nodes can be used for the mapping of the ring. In this case, the previously unused node now precedes the NOX.

### 4.2 Mapping of a Torus onto an RH

### 4.2.1 Mapping of a 2-D Torus onto an RII

The algorithm for the mapping of a $2-D$ torus onto the $R H(k, n)$ is an expanded version of the algorithm for ring mapping with $k=n$. The $2^{k-n}$ nodes contained in each $S B$ now form an entire row of the torus; this is accomplished by using the $R G C(k-n)$ to map a linear array with $2^{k-n}$ nodes onto the $S B$. Parallel connections between $S B$ 's now form columns of the torus. The address of each node in the torus is the pair of its Cartesian coordinates, i.e. its row and column addresses.

The encoded column addresses of torus nodes are then the ones assigned according to the $R G C(k-n)$. The mapping procedure for the ring is now used to form the sequence of rows. As each $S B$ forms a row, the problem is equivalent to mapping the ring with $2^{2^{n}+n}$ nodes (this is the total number of $S B$ 's) onto the $R H(n, n)$. Figure 4.2 shows the mapping of the $4 \times 8$ torus onto the $R H(3,1)$; all edges of the $R H$ are used in this case.

Figure 4.3 shows the mapping of the $2^{k-2} \times 60$ torus onto the $R H(k, 2)$. The ares illustrate the sequence of inter- $B B$ and inter- $S B$ edges selected by the mapping algorithm when starting with $S B 3$ of $B B 0$. The four encircled $S B$ addresses illustrate the unused $S B$ 's for this mapping.

With the above algorithm, a $2-D$ torus of size $)^{\prime} \times 2^{k-n}$ is mapped onto the $R H(k, n)$, where an upper bound on $Y$ is $2^{2^{n}+n}$. If $Y$ equals the upper bound then all nodes in the RH are used in an optimal mapping. The reason that $Y$ canot always attain its maximum value is the potential presence of some special SBs within the $B B$ 's, as stated in Theorem 4.3.

We can take one step further and count for a given $R H(k, n)$ how many pairs $0 \longleftrightarrow \delta$ of each kind we have. Since the special $S B$ 's can then be detemmed, the utilization of nodes in the $R H(k, n)$ can be calculated.

Theorem 4.4 Given the RH(k,n) there exist $2^{2^{n-5}} B B$ s with SOEASOX pairs of the form $0 \longleftrightarrow \delta$, for $\delta=1,2, \ldots, 2^{n}-2$ and $n>1$. and 4 pairs for $\delta=2^{n}-1$.

Proof. The proof applies mathematical induction with the RGC:(n).


Figure 4.2 Mapping of the $4 \times 8$ torus onto the RH(3.1)

Example 4.6 For the RH(k,3) we calculate the occurrences in each case. Special $S B$ 's are in boldface.
$0 \longleftrightarrow 1: 2^{7}=128$
$0 \longleftrightarrow 2: 2^{6}=64$
$0 \longleftrightarrow 3: 2^{5}=32$
$0 \longleftrightarrow 4: 2^{4}=16$
$0 \longleftrightarrow 5: 2^{3}=8$
$0 \longleftrightarrow 6: 2^{2}=4$
$0 \longleftrightarrow 7: 4$

The special SB's according to Theorem 4.3 are 3,5 and 6 . Hence, the number of unutilized $S B^{\prime}$ 's is $32+8+4=44$. The total number of $S B^{\prime}$ s is $2^{3}\left(\frac{S B}{B B}\right) \times 2^{8} B B=$ $2^{11}=2,048 S B^{\prime} s$. Therefore, Utilization $=\frac{2,048-44}{2,048}=0.9780197 .8 \%$

Corollary 4.2 There exists an optimal mapping of the $2-D$ torus of sian $2^{k-1} \times 2^{3}$ onto the RH(k, 1).

Proof. For $n=1$, according to Theorem 4.3 there are no special $S B$ 's, therefore all $S B$ 's are fully utilized.

Table 4.2 shows the utilization of $S B$ 's for the cases $n=1,2,3$ and 4 , which correspond to realistic systems. The results in Table 4.2 cleaty show that for $n>1$ the utilization is almost 1 and increases as $n$ increases.

### 4.2.2 Mapping of a $\mu-D$ Torus onto an RH

The mapping algorithm of Subsection 4.2 .1 is modified slightly here for the purpose of mapping the $\mu-D$ torus onto the $R H(k, n)$. The modification of the algorthm follows. In the mapping of the 2-D torus, all nodes in an $S B$ are used to form one dimension (i.e. a row) of the $2-\mathrm{D}$ torus and $S B$ to $S B$ parallel connections create the second dimension. It is evident that this second dimension mapping cannot change.


Figure 4.3 Mapping of the $2^{k-2} \times 60$ torus onto the $R H(k, 2)$

Table 4.2 Efficiency of torus mapping

| n | Total of BBs | SBs per BB | Total of SBs | Loss of SBs | Utilization |
| ---: | ---: | ---: | ---: | ---: | ---: |
| 1 | 4 | 2 | 8 | 0 | 1.000 |
| 2 | 16 | 4 | 64 | 4 | 0.937 |
| 3 | 256 | 8 | 2,048 | 44 | 0.978 |
| 4 | 65,536 | 16 | $1,048,576$ | 11,474 | 0.989 |

Therefore, we must extract more dimensions from each $S B$. Since the $S B$ is a regular cube, and since we have the theory for mapping a $\mu$-D torus onto a hypercube, this becomes an easy problem. It is obvious that the maximum number of dimensions that can be attained is equal to $k-n+1$.

### 4.3 Mapping of a Binary Tree onto an RH

The mapping of a complete (balanced) binary tree onto a regular hypercube can be carried out using either one of two approaches. The first approach uses a onc-to-one mapping while as a result of the second approach there exist hypercube nodes that simulate more than one binary tree node.

It is impossible to derive a one-to-one mapping of an $n$-level binary tree with $2^{n}-1$ nodes onto an $n$-cube with $2^{n}$ nodes, for all $n \geq 3$. The largest binary tree that can be mapped in one-to-one fashion onto an $n$-cube has $n-1$ levels and contains $2^{n-1}-1$ nodes $[3,12]$. Approximately half of the hypercube nodes are then ntilized. In fact, two ( $n-1$ )-level disjoint binary trees can be mapped simultancously, leaving just one unused hypercube node. We would like to examine the situation for the case of the $R H(k, n)$. Since the $R H(k, n)$ contains $2^{2^{n}}$ complete $k$-cubes ( $\left.1313^{\prime} s\right)$. then $2^{2^{n}+1}(k-1)$-level disjoint binary trees can be mapped simultaneously onto it. as each $B B$ can accommodate for two ( $k-1$ )-level binary trees.

An algorithm in [3] uses a single two-degree node as a child of the root and stretches it (i.e., creates a double root with a spacer node) to utilize all the nodes in the hypercube. The extra node is used only for communication between the root and one of its children. All edges have dilation one except for the one whose image contains the spacer node and therefore has dilation two. This way, an $n$-level (complete) binary tree is mapped onto an $n$-cube, and all nodes of the n-cube are used as a spacer node is also required.

The regular hypercube has symmetric structure and therefore any node can be chosen as the root of the binary tree when the algorithm in [3] is applied, as long as the same transformation is applied to all node addresses. Each $B B$ provides a $k$ - level binary tree with a spacer node, and pairs of binary trees can be linked together through their spacer nodes as presented in [3] (one inter- $B / 3$ edge, out of the $2^{k-n}$ edges that interconnect a pair of $B B$ 's, are used for this purpose). Therefore, a $(k+1)$-level binary tree with $2^{k+1}-1$ nodes is obtained from each pair of $B B \circ s$ for a total of $2^{2^{n-1}}$ binary trees; each such binary tree contains a single spacer node. The roots of these binary trees can then be interconnected in a bimary tree structure with maximum dilation $n+1$ because up to $n$ hops are required within a $B B$ in order to reach a $S B$ that implements connections in a specific dimension. Therefore, all lut. $2^{2 n-1}$ (spacer) nodes are utilized. Due to the latter binary tree, $2^{2 n-1}$ nodes are used twice in this mapping.

The resultant mapping has low dilation because practical values of $n$ are normally in the range from 1 to 3 . Therefore, RH's are also suitable for the emulation of binary trees.

## CHAPTER 5

## CONCLUSIONS

The main objective of the research presented in this thesis was to develop algorithms for embedding into the RH topology three topologies very frequently used for the development of parallel algorithms. These frequently used topologies are the ring. the torus, and the binary tree.

For the ring structure, the proposed mapping algorithm for the $R H(k, n)$, where $k>n$, achieves dilation one and all the nodes are utilized. Clearly this is an optimal mapping in all respects just as that for the regular hypercube. Taking into accoumt the fact that a realistic system will indeed have $k>n$. the above result is very important. Even for the case where $n=k$, a mapping with dilation one can be attained with a relatively very small number of unutilized nodes as shown in Table 4.! of Subsection 4.2.1. The case of $n=1$ is a special case in which no nodes romain unutilized. All nodes can be used in the case of $k=n$ if a dilation of two can be tolerated.

The mapping algorithm for the torus structure has turned ont to be an expanded version of the mapping algorithm for the ring structure for the case of $n=k$. For a mapping with dilation one only a small number of nodes romain unutilized for all $n>1$. For the mapping of a square torus with maximum ntilization of nodes, the following condition must be satisfied for the $R H(k, n): n-k=2^{2^{n}}$.

The problem of mapping the binary tree onto the RII was expected to be more challenging. The proposed algorithms are based on existing algorithms for the conventional hypercube. These algorithms were applied to RH/ building blocks which are complete $k$-cubes. The mapping of trees with maximum size was achieved by slightly modifying the algorithms and also connecting building blocks in pairs. Therefore, multiple trees are obtained, though smaller in size than those obtained
from a conventional hypercube with the same number of nodes. Although it seems that mapping of binary trees onto $R H$ 's is a more difficult problem than that of regular hypercubes, the fact that RH's permit the design of considerably larger systems is a major advantage. A larger system and a proper choice of $n$ and $k$ will result in desired tree sizes.

It has already been established that a $R H$ has significantly lower VLSI complexity compared to a regular hypercube with the same number of nodes. As a direct consequence of this, larger systems may be designed, thus achieving significant speedups. In addition it was shown that the RH maintains a low diameter and average distance with respect to its total number of nodes. Since it was also shown in this thesis that RH's permit the efficient mapping of frequently used wopogies. RH's are viable alternatives to the hypercube for massively parallol processing.

## REFERENCES

1. S.G. Ziavras, "On the Problem of Expanding Hypercube-Based Systems," J. Parallel Distrib. Computing 16, 1 (Sept. 1992), pp. 41-53.
2. S.G. Ziavras, "RH: A Versatile Family of Reduced Hypercube Interconnection Networks," To appear in IEEE Trans. Parallel Distributed Systems.
3. S.R. Deshpande and R.M. Jenevin, "Scalability of a Binary Tree on a Hypercube," Proc. International Conference on Parallel Processing. IEEE Computer Society, Silver Spring, MD, Aug. 1986, pp. 661-668.
4. K. Ghose and K.R. Desai, "The HCN: A Versatile Interconnection Network Based on Cubes," Proc. Supercomputing '89, IEEE Computer Society and ACM SIGARCH, Nov. 1989, pp. 426-435.
5. W.D. Hillis, The Connection Machine, MIT Press, Cambridge, MA, 1985.
6. C.T. Ho and S.L. Johnsson. "Spanning Balanced Trees in Boolean ("ubes." SIAM J. Sci. Stat. Comput. 10, 4 (July 1989). pp. $607-630$.
7. H.P. Katseff, "Incomplete Hypercubes," IEEE Trans. Compul. ('37. 5 May 1988), pp. 604-608.
8. F.P. Preparata and J. Vaillenin, "The Cube-Comected Cycles: A Versatile Network for Parallel Computation," Comm. ACM24, May 1981. pp. 300309.
9. H.P. Katseff, "Incomplete Hypercubes," IEEE Trans. Compul. (-37.5 (May 1988), p. 604-608.
10. N.F. Tzeng, H.L. Chen, and P.J. Chuang, "Embeddings in Incomplete Hypercubes," Proc. International Conference Parallel Processing, IEEE Computer Society, Silver Spring, MD, Aug. 1990, pp. 335-339 (Vol. I).
11. C. L. Seitz, "Concurent VLSI Architectures," IEEE Trans. (omput. (.33, 12. (Dec. 1984), pp. 1247-1265.
12. F.T. Leighton, Introduction to Parallel Algorithms and Architectures, Morgan Kaufmann Publ., San Mateo, (A, 1992, pp. 407-410.
13. T.F. Chan, and Y'. Saad, "Multigrid Algorithms on the Hypercule Multiprocessor," IEEE Trans. Comput. C-35, 11 (Aug. 1988), pp. 969-977.
14. S.L. Johnsson, "Communication Efficient Basic Linear Algebra ('omputation on Hypercube Architectures," J. Parallel Distrib. Compui. 4. 2 (Apr. 1987). pp. 133-172.
15. A.Y.Wu, "Embedding of Tree Networks into Hypercubes," J. Parallel Distrib. Comput. 2, 3 (Aug. 1985), pp. 238-249.

## REFERENCES

(continued)
16. T.H. Lai, and W. White, W. "Mapping Pyramid Algorithms into Hypercubes," J. Parallel Distrib. Comput. 9 (1990) pp. 42-54.
17. S.G. Ziavras and M.A. Siddiqui, "Pyramid Mappings onto Hypercubes for Computer Vision: Connection Machine Comparative Study," Concurency: Practice Experience, Vol. 5, No. 6, Sept. 1993, pp. 471-889.

