Embeddings into Crossed Cubes

Emad Abuelrub*, *Member, IAENG*

Abstract- **The hypercube parallel architecture is one of the most popular interconnection networks due to many of its attractive properties and its suitability for general purpose parallel processing. An attractive version of the hypercube is the crossed cube. It preserves the important properties of the hypercube and most importantly reduces the diameter by a factor of two. In this paper, we show the ability of the crossed cube as a versatile architecture to simulate other interconnection networks efficiently. We present new schemes to embed complete binary trees, complete quad trees, and cycles into crossed cubes.**

Index Terms- **binary trees, cycles, dilation, embedding, expansion, hypercubes, crossed cubes.**

I. NTRODUCTION

Hypercube architectures are loosely coupled parallel processors based on the binary cube network. Parallel computers based on the hypercube topology have gained widespread acceptance in parallel computing. Recently, many machines based on the hypercube have been designed and made commercially available. The hypercube offers a rich interconnection topology with large bandwidth, logarithmic diameter, simple routing and broadcasting of data, recursive structure that is naturally suited to divide and conquer applications, and the ability to simulate other interconnection networks with minimum overhead [2, 11, 14, 15, 20, 21]. Due to the popularity of the hypercube, many variations of the hypercube topology have been proposed to improve on its properties and computational power. El-Amaway and Latifi [7] proposed the folded hypercube to reduce the diameter and the traffic congestion with little hardware overhead. Preparata and Vuillemin [18] introduced the cube-connected cycles in which the degree of the diameter was reduced to 3. Efe [6] proposed an attractive version of the hypercube called the crossed cube, where preliminary studies proved that the crossed cube preserves many of the attractive properties of the hypercube and more importantly reduces the diameter by a factor of two [1, 4, 6, 8, 9, 22]. This implies that the crossed cube has an advantage over the hypercube when data communication is of major concern. It is well known that for parallel architectures, data communication cost dominates computation cost. Therefore, it is worthwhile to make comparative studies on crossed cubes and other interconnection networks, and explore the advantages provided by them. The problem of embedding one interconnection network into another is very important in the area of parallel computing for portability of algorithms across various architectures, layout of circuits in VLSI, and mapping logical data structures into computer memories.

*Department of Computer Science, Zarqa Private University, Jordan, email: abuelrub@zpu.edu.jo

The importance of binary trees comes from the fact that this class of structures is useful in the solution of banded and sparse systems, by direct elimination, and captures the essence of divide and conquers algorithms. Embedding binary trees into other interconnection networks attracted the attention of many researchers. Barasch *et al.* have considered embedding complete binary trees into generalized hypercubes [2], while Dingle and Sudborough considered simulation of binary trees and X-trees on pyramid networks [5].

Quad trees are becoming an important technique in the domain of image processing, computer graphics, robotics, computational geometry, and geographic information systems [3, 17, 19]. This hierarchical structure is based on the principle of recursive decomposition, which is similar to divide and conquer methods. Mapping quad trees into other interconnection networks attracted the attention of many researchers [14].

The problem of embedding rings or cycles into other interconnection networks has been studied by many researchers. It is well known that rings can be embedded into hypercubes using cyclic Gray codes [20]. Latifi and Zheng [12] generalized the cyclic Gray code method to embed rings into twisted cubes. On the other hand, other researchers addressed the problem of embedding rings into fault-free and faulty topologies [9, 12, 13, 14, 16, 20] or the Hamiltonicity of such structures in fault-free and faulty environments [1, 8, 10, 11].

The remainder of this paper is organized as follows. In section 2, we establish a few preliminary definitions and notations. Section 3 explains a scheme to embed complete binary trees into crossed cubes. Section 4 presents a recursive technique to embed complete quad trees. In section 5, we extend the Gray code scheme to embed cycles into crossed cubes. Finally, section 6 concludes the paper and discusses some future possible work.

II. DIFINITIONS AND NOTATIONS

In this paper, we use undirected graphs to model interconnection networks. Each vertex represents a processor and each edge a communication link between processors. The embedding of a guest graph $G = (V_G, E_G)$ into a host graph $H = (V_H, E_H)$ is an injective mapping f from V_G to V_H, where V_G, E_G and V_H, E_H are the vertex and edge sets of G and H, respectively, and where $|V_H| \geq |V_G|$. We consider a complete binary tree of height n-1, a complete quad tree of height n-1, and a cycle of size n, denoted CB_n, CQ_n, and C_n, respectively, as guest graphs and a crossed cube of dimension n, denoted XQ_n, as a host graph. Two cost functions, dilation and expansion often measure the quality of an embedding. If u and v are two nodes in G, then the distance from u to v, d = (u, v), is the length of the shortest path from u to v. The *dilation* D is the maximum distance in H between the images of

adjacent vertices of G, D = max {d(f(u), f(v)), where u–v $\in E_G$}. The *expansion* E is the ratio of the cardinality of the host vertex set to the cardinality of the guest vertex set, $E = |V_H| / |V_G|$. Minimizing each of these measurements has a direct implication on the quality of the simulation of the guest network by the corresponding host network. The dilation of an embedding measures how far apart neighboring guest processors are placed in the host network. Clearly, if adjacent guest processors are placed far apart in the host network, then there will be a significant degradation in simulation due to the long length of the communication path between them. The expansion of an embedding measures how much larger is the host network than the guest network during the simulation. We want to minimize expansion, as we want to use the smallest possible host network that has at least as many processors as in the guest network. In reality, we usually have a fixed size host network and we may have to consider many-to-one embedding for larger guest networks. When the size of the guest network is not equal to the size of the host network in terms of the number of processors, then we try to find the smallest host network that has at least as many processors as the guest network. Such a host network is referred to as the *optimal host network*.

A *hypercube* of dimension n, denoted by Q_n, is an undirected graph consisting of 2^n vertices labeled from 0 to 2^n-1 and such that there is an edge between any two vertices if and only if the binary representation of their labels differs in exactly one bit position. A *complete binary tree* of height n-1, denoted by CB_n, is an undirected graph consisting of 2^n-1 vertices and such that every vertex of depth less than n-1 has exactly two sons and every vertex of depth n-1 is a leaf. A *complete quad tree* of height n-1, denoted by CQ_n, is an undirected graph consisting of $(4^n-1)/3$ vertices and such that every vertex of depth less than n-1 has exactly four sons and every vertex of depth n-1 is a leaf. A *cycle* of size n, denoted C_n, is an undirected graph consisting of n vertices labeled from v_1 to v_n, such that node v_i is a neighbor with node $v_{(i+1) \bmod n}$, $1 \le i \le n$. A *path* (v_o, v_1, v_2, …, v_{n-1}) is a sequence of nodes such that each two consecutive nodes are adjacent. A cycle or a circuit is called a *Hamiltonian circuit* if it traverses every node of G exactly once.

The crossed cube is defined recursively as follows. Let G be any undirected labeled graph, then G^b is obtained from G by prefixing every vertex label with b. Two binary strings $x = x_1x_0$ and $y = y_1y_0$, each of length two, are *pair-related* if and only if (x, y) $\in$ {(00, 00), (10, 10), (01, 11), (11, 01)}. Now, we define a *crossed cube* of dimension n, denoted XQ_n, as an undirected graph consisting of 2^n vertices labeled from 0 to 2^n-1 and defined recursively as following:

1. XQ_1 is the complete graph on two vertices with labels 0 and 1.
2. For n > 1, XQ_n consists of two copies of XQ_{n-1} one prefixed by 0, XQ^0_{n-1}, and the other by 1, XQ^1_{n-1}. Two vertices $u = 0u_{n-2}...u_0 \in XQ^0_{n-1}$ and $v = 1v_{n-2}...v_0 \in XQ^1_{n-1}$ are adjacent, if and only if:
   a. $u_{n-2} = v_{n-2}$, if n is even, and
   b. For $0 \le i \le \lfloor (n-1)/2 \rfloor$, $u_{2i+1}\, u_{2i}$ and $v_{2i+1}\, v_{2i}$ are pair-related.

Figure 1 shows crossed cubes of dimension 3. XQ_n is constructed recursively based on the construction of XQ_{n-1} by pasting together a copy of XQ^0_{n-1} and the mirror image of XQ^1_{n-1}, then adding the appropriate links between the two copies according to the pair-related relationship. For clarity, we view the crossed cube XQ_n as a [2 x 2^{n-1}] grid. If the grid is partitioned horizontally into two equal parts, then all nodes above the horizontal line have a 0 as a prefix, while all nodes below the horizontal line have a 1 as a prefix.

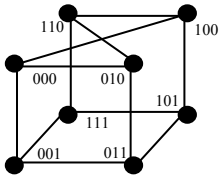


Figure 1: The crossed cube XQ_3.

III. EMBEDDING COMPLETE BINARY TREES

This section describes our scheme to embed a complete binary tree CB_n into a crossed cube XQ_n with dilation two and unit expansion. Our scheme is based on the inorder labeling to embed CB_n into XQ_n in a straight forward way. The inorder embedding is constructed by Algorithm Embedding Complete Binary Tree (ECBT).

Algorithm ECBT
Begin
Label the nodes of the complete binary tree based on the inorder traversal using binary representation.
Map each node of the complete binary tree to the node in the crossed cube with the corresponding binary representation.
End

Theorem 1: For all n, the inorder labeling of the complete binary tree embeds CB_n within the crossed cube XQ_n with dilation two.

Proof: Let β_k be the binary string of length k with 1 in all positions. For n 3, the inorder embedding is shown in Figure 2. For n > 3, we prove the theorem by induction on the height of the binary tree. Our induction basis is CB_3, a dilation two embedding of CB_3 into XQ_3 is shown in Figure 2. Assume the theorem is true for an embedding of CB_{n-1} into XQ_{n-1}. We now prove that the theorem is true for the embedding of CB_n into XQ_n. In XQ_n, consider the two subcubes XQ^0_{n-1} and XQ^1_{n-1}. By induction hypothesis, we can embed CB_{n-1} into XQ^0_{n-1} and XQ^1_{n-1}, with dilation two. Since the number of nodes in CB_{n-1} is less than the number of nodes in XQ_{n-1} by one, then XQ^0_{n-1} and XQ^1_{n-1} contain two extra-unused nodes located at addresses $01\beta_{n-2}$ and $11\beta_{n-2}$, respectively. We can use the extra-unused node in XQ^0_{n-1}, the CB_{n-1} of XQ^0_{n-1}, and the CB_{n-1} of XQ^1_{n-1} to construct the complete binary tree CB_n with 2^n-1 nodes. Next, we prove that the dilation of this embedding is two. We again use the routing algorithm of [12] to show that the length of the shortest path from the root of CB_n to any of its children is of length two. Let r~lr be the shortest path from the root r of CB_n to the root lr of the left complete

binary subtree LCB_{n-1} and r~rr be the shortest path from the root r of CB_n to the root rr of the right complete binary subtree RCB_{n-1}. r will appear at address $01\beta_{n-2}$, lr at address $00\beta_{n-2}$, and rr at address $10\beta_{n-2}$. Notice that r, lr, and rr are identical except for the left most two bits. By using the routing algorithm, the shortest paths from $01\beta_{n-2}$ to $00\beta_{n-2}$ and from $01\beta_{n-2}$ to $10\beta_{n-2}$ are of length two. So, the dilation of this embedding is two. □

As an illustration to the resulted embedding, in the lowest level, each edge from a left child to its parent is mapped to the corresponding crossed cube edge between the images of the two nodes, while the edge between a right child to its parent is mapped to a path of length two, from the right child to the left child and from the left child to the parent. In the higher level, each edge from a left child, or a right child, to its parent is mapped to the corresponding crossed cube edge between the images of the two nodes. In all higher levels, each edge from a left child, or a right child, to its parent is mapped to a path of length two. Notice that the inorder embedding is very simple and straight forward.

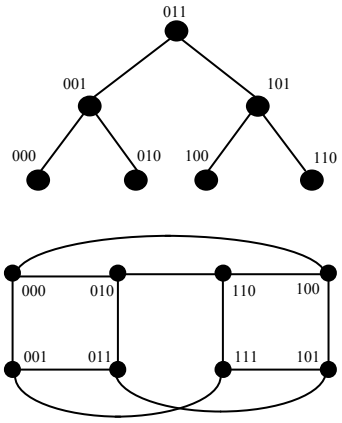


Figure 2: The inorder embedding of CB_3 into XQ_3.

IV. EMBEDDING QUAD TREES

This section describes our recursive scheme to embed a complete quad tree CQ_n into its optimal crossed cube XQ_{2n-1} with dilation two and unit expansion. We proceed in four steps. In the first step, CQ_n is decomposed into a four complete quad sub trees; a left complete quad sub tree ACQ_{n-1} with root a, a left middle complete quad sub tree BCQ_{n-1} with root b, a right middle complete quad sub tree CCQ_{n-1} with root c, a right complete quad sub tree DCQ_{n-1} with root d, and a root r. In the second step, XQ_{2n-1} is decomposed into four sub cubes XQ^{00}_{2n-3}, XQ^{01}_{2n-3}, XQ^{11}_{2n-3}, and XQ^{10}_{2n-3}. In the third step, ACQ_{n-1} is embedded into XQ^{00}_{2n-3}, BCQ_{n-1} is embedded into XQ^{01}_{2n-3}, CCQ_{n-1} is embedded into XQ^{11}_{2n-3}, DCQ_{n-1} is embedded into XQ^{10}_{2n-3}, and the root r is embedded into one of the unused nodes in XQ^{00}_{2n-3}. In the last step, we construct CQ_n by finding the paths r~a, r~b, r~c, and r~d, each of at most length two. The embedding process is continued recursively by decomposing the complete quad sub trees and the crossed sub cubes, repeating the above steps, until we reach the leaves of the complete quad tree. At the bottom level of the complete quad tree, each complete quad sub tree with 5 nodes is mapped into a crossed sub cube of dimension 3. Next, we present Algorithm Embed Complete Quad Tree (ECQT) that uses a recursive divide and conquers technique to embed a complete quad tree CQ_n into its optimal crossed cube XQ_{2n-1}.

Algorithm ECQT
Let δ_i be the binary string of length n with a 1 in position i and 0 in all other positions, θ_k be the binary string of length k with 0 in all positions, and $\oplus$ be the xor operator.
Begin
Decompose CQ_n to ACQ_{n-1}, BCQ_{n-1}, CCQ_{n-1}, DCQ_{n-1}, and r.
Decompose XQ_{2n-1} to XQ^{00}_{2n-3}, XQ^{01}_{2n-3}, XQ^{11}_{2n-3}, and XQ^{10}_{2n-3}.
Map the quad sub trees into the crossed sub cubes as follows:
a. Embed ACQ_{n-1} into XQ^{00}_{2n-3}, BCQ_{n-1} into XQ^{01}_{2n-3}, CCQ_{n-1} into XQ^{11}_{2n-3}, and DCQ_{n-1} into XQ^{10}_{2n-3}. a, b, c, and d will appear at addresses $000\theta_{2n-4}$, $010\theta_{2n-4}$, $110\theta_{2n-4}$, and $100\theta_{2n-4}$,, respectively.
b. Translate the embeddings in XQ^{00}_{2n-3} and XQ^{10}_{2n-3} by complementing the $(2n-3)^{th}$ bit of each node. Formally, if a tree node was mapped to address x then after the translation it will appear at address $x \oplus \delta_{2n-3}$. After the translation, the left root a and the right root d will appear at addresses $001\theta_{2n-3}$ and $101\theta_{2n-4}$, respectively. Therefore, the final position of a, b, c, and d are $001\theta_{2n-4}$, $010\theta_{2n-4}$, $110\theta_{2n-4}$, and $101\theta_{2n-4}$,, respectively.
c. Map the root r into the node with label 0 in XQ^{00}_{2n-3}.
Construct CQ_n from ACQ_{n-1}, BCQ_{n-1}, CQ_{n-1}, DCQ_{n-1}, and r by finding the four paths r~a, r~b, r~c, and r~d. The edges r-a and r-b of CQ_n are mapped to paths of length one in XQ_{2n-1}, while the edges r-c and r-d are mapped to paths of length two. The shortest paths from r to c and from r to d are $000\theta_{2n-4}$ - $010\theta_{2n-4}$ - $110\theta_{2n-4}$ and $000\theta_{2n-4}$ - $100\theta_{2n-4}$ -$101\theta_{2n-4}$,respectively.
End

Theorem 2: For all n, Algorithm ECQT maps the complete quad tree CQ_n within the crossed cube XQ_{2n-1} with dilation two and unit expansion.
Proof: For n = 1, CQ_1 can be easily embedded into XQ_1. For n = 2, the existence of a mapping with dilation two is shown in Figure 4. Since the crossed cube is vertex symmetric, then we can relabel the crossed cube and any of the corner nodes of the sub cube XQ_3 can be the root.

Figures 4-b and 4-c show two different kinds of possible embeddings that might be used during the embedding process. For n > 2, we prove this by induction on the height of the complete quad tree CQ_n. Our induction basis is CQ_3, a dilation two mapping of CQ_3 into XQ_5 is shown in Figure 5. Note that the figure shows only the positions of the roots, as the rest of the embedding follows directly from Figure 4. Assume the theorem is true for a mapping of CQ_{n-1} in XQ_{2n-3}. Now, we prove that the theorem is true for the embedding of CQ_n in XQ_{2n-1}. In XQ_{2n-1}, consider the four sub cubes XQ^{00}_{2n-3}, XQ^{01}_{2n-3}, XQ^{11}_{2n-3}, and XQ^{10}_{2n-3}. By induction hypothesis, there exist a dilation two mapping from CQ_{n-1} to XQ^{00}_{2n-3}, XQ^{01}_{2n-3}, XQ^{11}_{2n-3}, and XQ^{10}_{2n-3}. Since the number of nodes in CQ_{n-1} is less than the number of nodes in XQ_{2n-3}, then XQ^{00}_{2n-3}, XQ^{01}_{2n-3}, XQ^{11}_{2n-3}, and XQ^{10}_{2n-3} contain extra unused nodes. Now, we can use the unused node with label 0 in XQ^{00}_{2n-3}, the CQ_{n-1} of XQ^{00}_{2n-3}, the CQ_{n-1} of XQ^{01}_{2n-3}, the CQ_{n-1} of XQ^{11}_{2n-3}, and the CQ_{n-1} of XQ^{10}_{2n-3} to construct the complete quad tree CQ_n. To prove that the dilation of this mapping is two, we need to show that the length of the shortest path from the root r to any of its four children is at most two. Clearly, the length of the paths r~a and r~b is one since they are mapped directly to edges in the crossed cube. Let r~c be the shortest path from the root r of CQ_n to the root c of the right middle complete quad sub tree CCQ_{n-1} and r~d be the shortest path from the root r of CQ_n to the root d of the right complete quad sub tree DCQ_{n-1}. r will appear at address $000\theta_{2n-4}$, c will appear at address $110\theta_{2n-4}$, and d will appear at address $101\theta_{2n-4}$. Notice, that if we group the addresses of r, c, and d into pairs of bits, from right to left, then they are pair-related except for the left most three bits. By using the routing algorithm of [6], the shortest path from $000\theta_{2n-4}$ to $110\theta_{2n-4}$ is $000\theta_{2n-4}$ - $010\theta_{2n-4}$ -$110\theta_{2n-4}$ and from $000\theta_{2n-4}$ to $101\theta_{2n-4}$ is $000\theta_{2n-4}$ - $100\theta_{2n-4}$ -$101\theta_{2n-4}$. So, the length of the paths r~c and r~d are two. Therefore, the dilation of this embedding is two. □

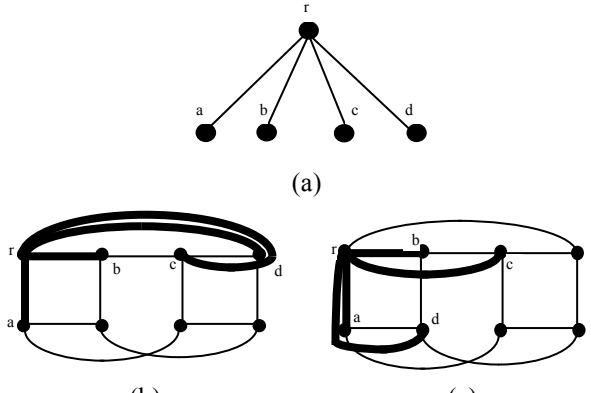


Figure 4: Embedding CQ_1 into the sub cube XQ_3.

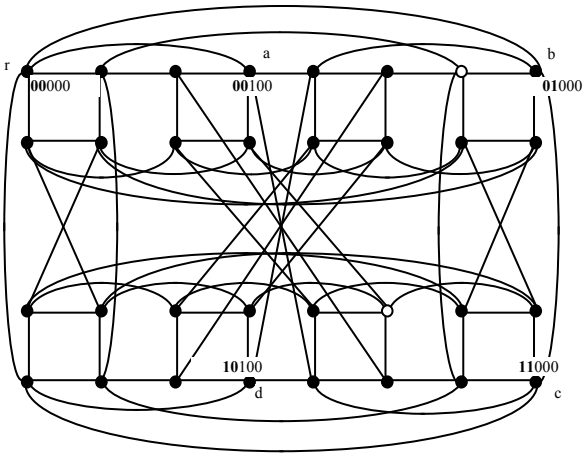


Figure 5: Embedding CQ_3 into XQ_5.

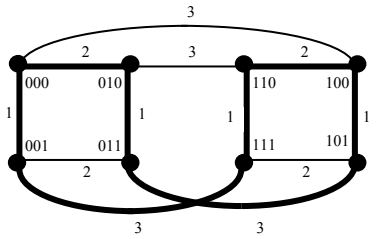
V. EMBEDDING CYCLES

Given a cycle C_{2n} with 2^n nodes, consider the problem of assigning the cycle nodes to the nodes of the crossed cube XQ_n such that adjacency is preserved. Now, given any two adjacent nodes in the cycle, their images by this embedding should be neighbors in the crossed cube through some dimension i, where $1 \le i \le n$. We can view such an embedding as a sequence of dimensions crossed by adjacent nodes. We call such a sequence the *embedding sequence*, denoted by ES = (d_1, d_2, ..., d_{2n}), where $d_i \in \{1, ..., n\}$ for all $1 \le i \le 2^n$. Figure 6 shows two different embeddings of the cycle C_{23} into the crossed cube XQ_3. It is more convenient to view the embedded cycle as will as the crossed cube in the way shown in Figure 6. The embedding sequence of C_{23} is ES = (1, 3, 1, 2, 1, 3, 1, 2). For example, in Figure 6-a, notice that nodes 000 and 001 are connected by a link through dimension 1, 001 and 111 are connected by a link through dimension 3, 111 and 110 are connected by a link through dimension 1, 110 and 100 are connected by a link through dimension 2, and so on. The embedding sequence ES can be generated using Algorithm ES.

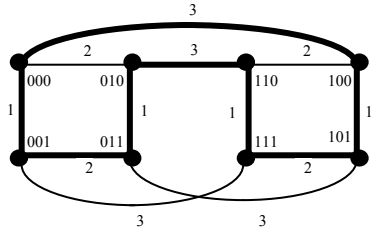
Algorithm ES
Let n be the dimension of the crossed cube and the vertical bar be the concatenation operator.
Begin
    $ES \leftarrow 1$
    For $i \leftarrow 3$ *to n do*
        $ES \leftarrow ES \,/\, i \,/\, ES$
    $ES \leftarrow ES \,/\, 2 \,/\, ES \,/\, 2$
End

The embedding sequence is generated by applying Algorithm ES on n, where n is the dimension of the crossed cube. The number of nodes in the crossed cube is equal to the number of nodes in the embedded cycle, which is 2^n nodes. Thus, the embedding sequence of the cycle C_{24} is ES = (1, 3, 1, 4, 1, 3, 1, 2, 1, 3, 1, 4, 1, 3, 1, 2) and the embedding sequence of the cycle C_{25} is ES = (1, 3, 1, 4, 1, 3, 1, 5, 1, 3, 1, 4, 1, 3, 1, 2, 1, 3, 1, 4, 1, 3, 1, 5, 1, 3, 1, 4, 1, 3, 1, 2). The embedding sequence corresponds to the extended binary-reflected Gray code embedding of a cycle into a crossed cube. The binary-reflected Gray code is the

most common technique to embed a cycle into a fault-free hypercube. Notice that the same embedding sequence may result in different embeddings of C_2n into XQ_n depending on the crossed cube node that initiates the cycle construction. Among all different embeddings, we are interested in one kind. The embedding when the node that initiates the cycle construction in the crossed cube is the upper leftmost node, node with label 0. This will not violate the generalization of the technique since the crossed cube is node and vertex symmetric, which means that we can relabel the nodes, where any node can be labeled as node 0, and hence initiates the construction of the cycle. In Figure 6-a, the cycle is initiated by node 000, while in Figure 6-b the cycle is initiated by node 001.



(a) Node 000 initiating the embedding sequence.



(b) Node 001 initiating the embedding sequence.
Figure 6: The embedding sequence.

Theorem 3: For every n, Algorithm ES will generate the embedding sequence to construct a cycle of size 2^n in a crossed cube of dimension n.
Proof: We prove this by induction on the dimension of the crossed cube. Our induction basis is XQ_2, it is trivial that a cycle of size 4 can be easily constructed in XQ_2 using the embedding sequence ES = (1, 2, 1, 2). Assume the theorem is true for the construction of a cycle of size 2^{n-1} in a crossed cube of dimension n-1. We now prove that the theorem is true for the construction of C_2n in XQ_n. Let G be any undirected labeled graph, then G^b is obtained from G by prefixing every vertex label with b. Consider the two crossed sub cubes XQ^0_{n-1} and XQ^1_{n-1}. By induction hypothesis, we can construct a cycle of size 2^{n-1} in both XQ^0_{n-1} and XQ^1_{n-1}. Let their embedding sequence be ES = $S_{n-1} | 2 | S_{n-1} | 2$, where, S_n is a sequence of dimensions recursively defined as follows: $S_2 = 1$ and $S_{n-1} = S_{n-2} | n | S_{n-2}$. Now, we combine two cycles, each of size 2^{n-1}, to come up with a cycle of size 2^n. This is done by replacing the first link that goes through dimension 2 of the first cycle and the second link that goes through dimension 2 of the second cycle by two links that go through dimension n. The embedding sequence of the new cycle C_2n is ES = $S_{n-1} | n | S_{n-1} | 2 | S_{n-1} | n | S_{n-1} | 2 = S_n | 2 | S_n | 2$, which is the same embedding sequence generated by Algorithm ES. □

Next, we present Algorithm Hamiltonian Circuit (HC) that uses a recursive divide and conquers technique to embed a cycle C_2n into a crossed cube XQ_n.

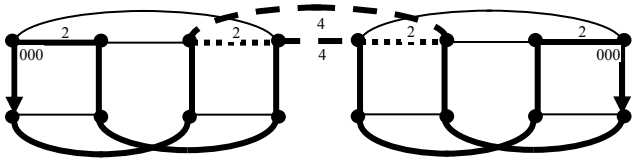
Algorithm HC
Begin
Partition XQ_n into 2^{n-3} disjoint crossed cubes, each of dimension 3.
Embed the cycle C_23 into each sub cube using the embedding sequence ES = (1, 3, 1, 2, 1, 3, 1, 2).
Connect the 2^{n-3} cycles, each of size 8, through the upper, or lower, links to come up with a cycle of size C_2n.
End

Theorem 4: For every n, Algorithm HC will embed a Hamiltonian cycle of size 2^n in a crossed cube of dimension n.
Proof: We prove this by induction on the dimension of the crossed cube. Our induction basis is XQ_3, the embedding of a cycle of size 2^3 into a crossed cube of dimension 3 is shown in Figure 6. Assume the theorem is true for the construction of a cycle of size 2^{n-1} in a crossed cube of dimension n-1. We now prove that the theorem is true for the construction of C_2n in XQ_n. Consider the two crossed sub cubes XQ^0_{n-1} and XQ^1_{n-1}. By induction hypothesis, we can construct a cycle of size 2^{n-1} in both XQ^0_{n-1} and XQ^1_{n-1}. Now, we combine two cycles, each of size 2^{n-1}, to come up with a cycle of size 2^n. This is done by replacing the first link that goes through dimension 2 of the first cycle constructed in XQ^0_{n-1} and the first link that goes through dimension 2 in of the second cycle constructed in XQ^1_{n-1} by links that go through dimension n. Note the use of the upper links of dimension n when the embedding sequence is generated by node 0, while the lower crossed links of dimension n are used when the embedding sequence is generated by node 1, as shown in Figure 7. □

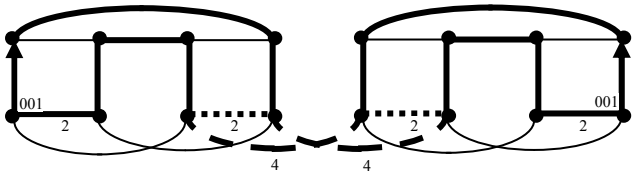
VI. CONCLUSIONS

The problem of embedding one interconnection network into another is very important in the area of parallel computing for portability of algorithms across various architectures, layout of circuits in VLSI, and mapping logical data structures into computer memories. The importance of trees comes from the fact that this class of structures is useful in the solution of banded and sparse systems, by direct elimination, and captures the essence of divide and conquers algorithms. The important of cycles and rings comes from the fact that it is the task graph of many basic operations in parallel computing. The hypercube is one of the most popular interconnection networks due to many of its attractive properties and its suitability for general purpose parallel processing. An attractive version of the hypercube is the crossed cube. It preserves many properties of the hypercube and most importantly reduces the diameter by a factor of two. This paper has presented different schemes to show the ability of the crossed cube as a versatile architecture to simulate other interconnection networks efficiently. We present new schemes to embed complete binary trees, complete quad trees, and cycles into crossed cubes. A good problem will be to improve the dilation on embedding trees into crossed

cubes. Another interesting problem is to generalize the schemes to embed trees and cycles in the presence of faults.



(a) Using upper links when ES initiated by 000.



(b) Using lower links when ES initiated by 001.

Figure 7: The recursive construction of the cycle C_{2^n} in a fault-free environment.

REFERENCES

[1] E. Abuelrub, "The Hamiltonicity of Crossed Cubes in the Presence of Faults," *Engineering Letters*, vol. 16, no. 3, pp. 453-459, 2008.

[2] L. Barasch, S. Lakshmivarahan, and S. Dhall, "Embedding Arbitrary Meshes and Complete Binary Trees in Generalized Hypercubes," *Proceedings of the 1st IEEE Symposium on Parallel and Distributed Processing*, pp. 202-209, 1989.

[3] A. Bardera, M. Feixas, I. Boada, J. Rigau, and M. Sbert, "Medical Image Registration Based on BSP and Quad-Tree Partitioning", *Proceedings of WBIR'2006*, pp. 1-8, 2006.

[4] C. Chang, T. Sung, and L. Hsu, "Edge Congestion and Topological Properties of Crossed Cubes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 1, pp. 64-80, 2006.

[5] A. Dingle and I. Sudborough, "Simulating Binary Trees and X-Trees on Pyramid Networks," *Proceedings of the 1st IEEE Symposium on Parallel and Distributed Processing*, pp. 210-219, 1989.

[6] K. Efe, "The Crossed Cube Architecture for Parallel Computation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 5, pp. 513-524, 1992.

[7] A. El-Amaway and S. Latifi, "Properties and Performance of Folded Hypercubes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 2, no. 1, pp. 31-42, 1991.

[8] J. Fan, X. Lin, and X. Jia, "Non-Pancylicity and Edge-Pancyclicity of Crossed Cubes," *Information Processing Letters*, vol. 93, no. 3, pp. 133-138, 2005.

[9] J. Fan, X. Lin, and X. Jia, "Optimal Path Embedding in Crossed Cubes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 12, pp. 1190-1200, 2005.

[10] J. Fu and G. Chen, "Hamiltonicity of the Hierarchical Cubic Network," *Theory of Computer Systems*, vol. 35, pp. 59-79, 2008.

[11] H. Keh and J. Lin, "On Fault-Tolerance Embedding of Hamiltonian Cycles, Linear Arrays, and Rings in a Flexible Hypercube," *Parallel Computing*, vol. 26, no. 6, pp. 769-781, 2000.

[12] S. Latifi and S. Zheng, "Optimal Simulation of Linear Array and Ring Architectures on Multiply-Twisted Hypercube," *Proceedings of the 11th International IEEE Conference on Computers and Communications*, 1992.

[13] S. Lee and H. Ho, "A 1.5 Approximation Algorithm for Embedding Hyperedges in a Cycle," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 6, pp. 481-487, June 2005.

[14] T. Leighton, Introduction to Parallel Algorithms and Architecture: Arrays, Trees, Hypercubes, *Morgan Kaufmann*, 1992.

[15] J. Lin, "Embedding Hamiltonian Cycles, Linear Arrays, and Rings in a Faulty Supercube," *International Journal of High Speed Computing*, vol. 11, no. 3, pp. 189-201, 2000.

[16] M. Ma and J. Xu, "Panconnectivity of Locally Twisted Cubes," *Applied Mathematical Letters*, vol. 17, no. 7, pp. 674-677, 2006.

[17] T. Markas and J. Reif, "Quad Tree Structures for Image Compression Applications", *Information Processing Letters*, vol. 28, no. 6, pp. 707-722, 1992.

[18] F. Preparata and J. Vuillemin, "The Cube-Connected Cycles: A Versatile Network for Parallel Computation," *Communications of the ACM*, vol. 24, no. 5, pp. 3000-309, 1981.

[19] L. Topi, R. Parisi, and A. Uncini, "Spline Recurrent Neural Network for Quad-Tree Video Coding," *Proceedings of WIRN'2002*, pp. 90-98, 2002.

[20] Y. Saad and M. Schultz, "Topological Properties of the Hypercube," *IEEE Transactions on Computers*, vol. 37, no. 7, pp. 867-872, July 1988,

[21] L. Youyao, "A Hypercube-based Scalable Interconnection Network for Massively Parallel Computing," *Journal of Computers*, vol. 3, no. 10, October 2008.

[22] Q. Zhu, J. Hou, and M. Xu, "On Reliability of the Folded Hypercubes," *Information Science*, vol. 177, no. 8, pp. 1782-1788, 2008.