2004

A heuristic on the rearrangeability of shuffle-exchange networks

Katherine Yvette Alston

A HEURISTIC ON THE REARRANGEABILITY OF SHUFFLE-EXCHANGE NETWORKS

A Thesis

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Katherine Yvette Alston

June 2004

A HEURISTIC ON THE REARRANGEABILITY OF

SHUFFLE-EXCHANGE NETWORKS

A Thesis

Presented to the

Faculty of

California State University,

San Bernardino

---

by

Katherine Yvette Alston

June 2004

Approved by:

[redacted] 6/2/04

Dr. Kay Zemoudeh, Chair, Computer Science Date

[redacted]

Dr. Ernesto Gomez

[redacted]

Dr. Keith Schubert

ABSTRACT

An interconnection network that passes all N! permutations, when $N=2^n$, in one pass through the network is defined as rearrangeable. No formal proof has been developed to show that (2n-1) stages of the shuffle-exchange (SE) network are sufficient to pass all permutations.

The heuristic developed for the SE network relies on the use of the topologically equivalent baseline (BL) network concatenated with the cube-connected (CC) network. This thesis establishes a control heuristic for setting the states of the switching elements for arbitrary permutations for a 7-stage (2n-1) BL.CC network with N=16 (2^n) inputs. The conditions and requirements, which were discovered, will help serve as a foundation for future work on the proof of the rearrangeability of the SE network.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Kay Zemoudeh, for continuously working with me and guiding me towards the completion of this thesis. His patience, encouragement, and extensive knowledge are very much appreciated. Thank-you to my committee members for their support. I would also like to thank Victor Sciortino for his senior project work.

DEDICATION

To my loving and supportive family:

Clynton, Chanté, and Clynton Jr.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

OVERVIEW

Introduction

The algorithms, which control network routing, are specific to the network because the algorithms are designed to take advantage of that network's topology. The "goodness" of a network includes such criteria as a simple routing algorithm and a simple routing algorithm would increase the use of the SE network.

No formal proof has been developed to show that (2n-1) stages of the SE network are rearrangeable. Subsequently, there is no simple routing algorithm that allows one to set the states of the switches and pass all permutations in (2n-1) stages. This thesis provides insight into the required conditions to establish the proof that (2n-1) stages of the SE network are rearrangeable.

Definition of Terms

1. Binary Switching Element: A (2 X 2) switch that has two inputs and two outputs as well as two possible switch settings, through or cross. If the setting is "through", the upper and lower inputs go to the

upper and lower outputs respectively. If the setting is "cross", the upper and lower inputs go to the lower and upper outputs respectively. See Figure 1.

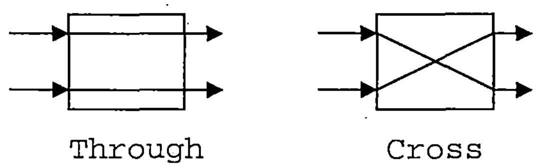


Figure 1. Switch Settings

2. Line Numbers (LN): The input lines to each binary switch. See Figure 2.

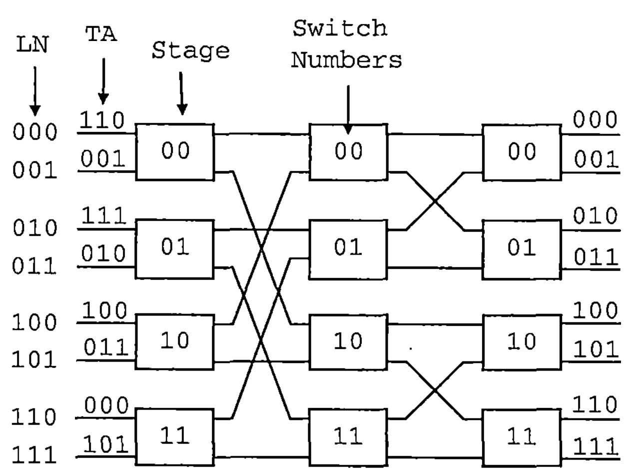


Figure 2. Cube-Connected Network

3. Target Address (TA): Identifies the destination processor. See Figure 2.

4. Stage: A column consisting of 2^{n-1} binary switches for $N=2^n$ inputs. See Figure 2.

5. Routing: Moving information, or transmitting a message, across a network from a source to a destination. See Figure 3 for an example of a CC routing algorithm applied to a permutation. The arrows point to the position of the control bit.

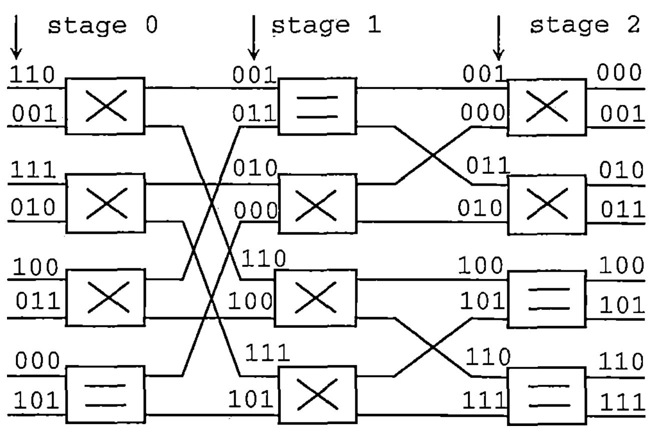
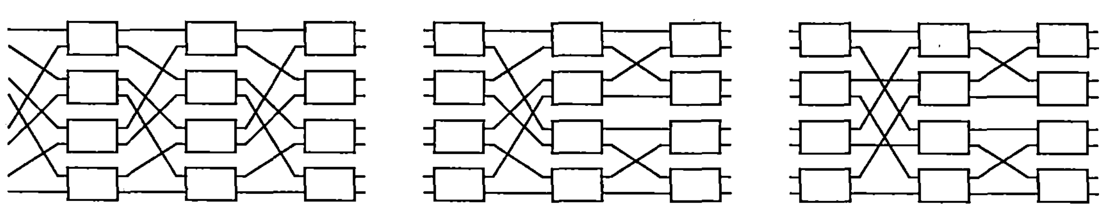


Figure 3. Cube-Connected Network Routing

6. Switching Network: Switching elements are used to establish time variant paths among processors. All processors are connected to both the inputs and outputs of one or more switching elements. The switching elements, based on the target addresses, decide on the connections that must be made to

establish a path. A CC network is an example of a switching network.

7. Multistage Interconnection Networks (MINs): A MIN consists of a number of binary switching elements arranged in several stages such that the output lines of one stage are the input lines of the next stage. There are $N=2^n$ inputs and outputs, N/2 switches per stage, and $O(\log_2 N)$ stages. The input lines are numbered from 0 to N-1 from top to bottom. The connection between stages is controlled by some interconnection function. Different MINs are constructed based on changing the interconnection function that exists between the stages. Figure 4 shows three examples of MINs: SE, BL, and CC. A MIN of size 2 is a switch.



SE Network BL Network CC Network

Figure 4. Multistage Interconnection Networks

8. Output Contention: More than one input attempts to send data to the same output.

9. Blocking: When there is no output contention and an input sent to a particular output blocks another input from going to some output. In a blocking network, data cannot flow on all connections simultaneously.

10. Non-blocking: Any input can go to any output without interfering with another input going to an output. Data can flow on all connections simultaneously.

11. Rearrangeablity: An interconnection network that passes all N! permutations in one pass through the network is called rearrangeable.

12. Shuffle-Exchange Networks: Identical stages in which there is a perfect shuffle of the input lines followed by an exchange between the switches. See Figure 5.

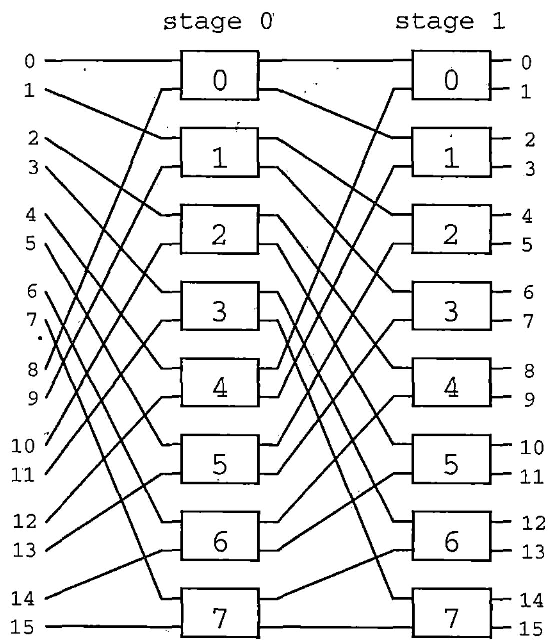


Figure 5. A Two-Stage Shuffle-Exchange Network

Purpose

The control heuristic provided in this thesis was developed in the quest to prove the rearrangeability of the (2n -1) stage SE network for $N=2^n$ inputs. It provides a simple routing heuristic that allows one to set the states of the switches for N=16 inputs and pass most permutations in (2n-1)=7 stages. This thesis documents minimum requirements that can be used to develop a simple routing algorithm for SE networks.

Scope

This thesis establishes a heuristic. It provides a method or tool for passing most permutations, but not all. Those who work with SE networks and have a need to route 16 (2^4) inputs benefit from using this control heuristic. Some permutations can be quickly and easily routed through the network. Additionally, the user is able to gain an understanding as to the constraints and conditions which must be met for routing the more difficult permutations. Whenever an exact solution is not generated, an approximate solution is made available. One can start with an approximate solution and generate an exact solution through the use of iterations.

Significance

A 5-stage SE network with N=8 inputs has been shown to be rearrangeable [10]. This work for 16 inputs will help further the cause for the use of SE networks in more routing platforms. The lack of a good (minimum stage) non-blocking routing algorithm for SE networks that scales well hinders its use in some applications. Since unique paths and the chance of blocking exist in current SE networks of size $N \geq 16$, a simple routing algorithm that constructs a non-blocking SE network in (2n-1) stages will be useful. The techniques used in generating the switch settings to route 16 inputs could be extended to larger 2^n data sets because all of the rules and constraints are applicable regardless of the value of n.

The ability of a SE network to send all inputs to all outputs simultaneously if the network is non-blocking increases the speed of the network for parallel processing and other data transfer applications. All work towards establishing a non-blocking SE network contributes to the development of faster, more efficient networks.

Limitations

There are several ways to set the switches and route an arbitrary permutation. This control heuristic provides one way to set the switches to route a particular permutation. All possible solutions are not generated. This control heuristic is specific to 16 inputs and would have to be modified to scale to larger sets of permutations. Permutations are limited to the form of 2^n and those outside of this format do not work with this methodology. The approach taken in this thesis was to work with 16 inputs because the data set was manageable.

CHAPTER TWO

BACKGROUND

History of Shuffle-Exchange Networks

SE networks were initially proposed by Stone and the proof of their rearrangeablity has challenged researchers for decades [1]. The lower bound of (2n-1) stages was established as necessary to pass any N! permutation for $N=2^n$ through a SE network. Permutations were shown that could not be realized in fewer than (2n-1) stages and these permutations provided the proof for this lower bound [2]. However, no formal proof has ever been developed to show that a (2n-1) stage network is sufficient to pass all permutations.

SE networks have been studied extensively in parallel processing due to their efficient interconnection scheme [3][4][5]. A SE network with N input lines and $\log_2 N$ stages is called an Omega network, Figure 6. An Omega network is capable of passing some important classes of permutations that are useful in parallel processing. Most importantly, the Omega network can be controlled by a simple routing algorithm. Unfortunately, when N is large, the network can only perform a small fraction of the N! possible permutations [2].

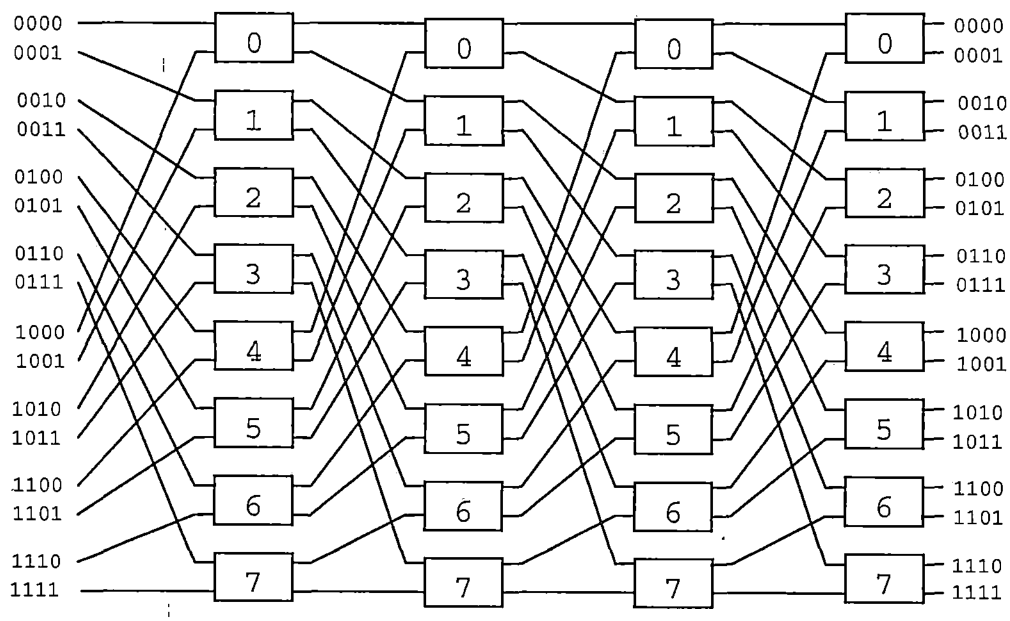


Figure 6. 16-Input Omega Network

Supporting Research

Algorithms that define a non-blocking SE network have been improving in terms of the minimum number of stages required. Following are some established proofs to date.

Stone developed an algorithm for a non-blocking SE network. It required $(\log_2 N)^2$ stages [1].

In 1975, Benes conjectured that a (2n-1) stage SE network was rearrangeable [6]. He established the well-known (2n-1) lower bound but never proved the sufficiency of (2n-1) stages to pass all permutations.

Siegel developed an algorithm for performing arbitrary permutations on a single-stage shuffle/exchange network in $3/2(\log_2 N)^2 - (\log_2 N)/2$ passes [7]. Parker subsequently improved this bound to $3\log_2 N$ stages [8]. Unfortunately, Parker did not specify a control algorithm for determining the states of the switching elements.

The best-known rearrangeable SE algorithm was established by Wu and Feng [9]. They observed that $(3\log_2 N - 1)$ stages are sufficient for rearrangeability of the SE network. Another significant accomplishment of Wu and Feng is that they showed how to compute the switch settings for arbitrary permutations.

CHAPTER THREE

NETWORKS

Design

Shuffle-Exchange

Switch Connections. Each identical stage of the SE network consists of a perfect shuffle of the input lines followed by switching elements. Figure 7 depicts the switch connections, which show that the output lines from switches in one stage are input to two different switches in the next stage. To determine which switch in a stage is connected to which switch in the next stage, left shift one place the switch number in the previous stage and add a "0" or "1" to the end.

If a switch is located in the upper half, its output lines connect to the upper inputs of the switches in the next stage. If a switch is located in the lower half, its output lines connect to the lower inputs of the switches in the next stage.

Alternate Method. Another way to determine the SE switch connections is to start with the switches in the next stage. For each switch, right shift one place its switch number and add a "0" or "1" to the beginning. The two resulting numbers are the switches in the previous stage to which it is connected.

The "0" in the beginning indicates that the upper input line comes from a switch in the upper half of the previous stage. The "1" in the beginning indicates that the lower input line comes from a switch in the lower half of the previous stage. Besides understanding of how the switches are connected it is important to know how the line numbers are determined.

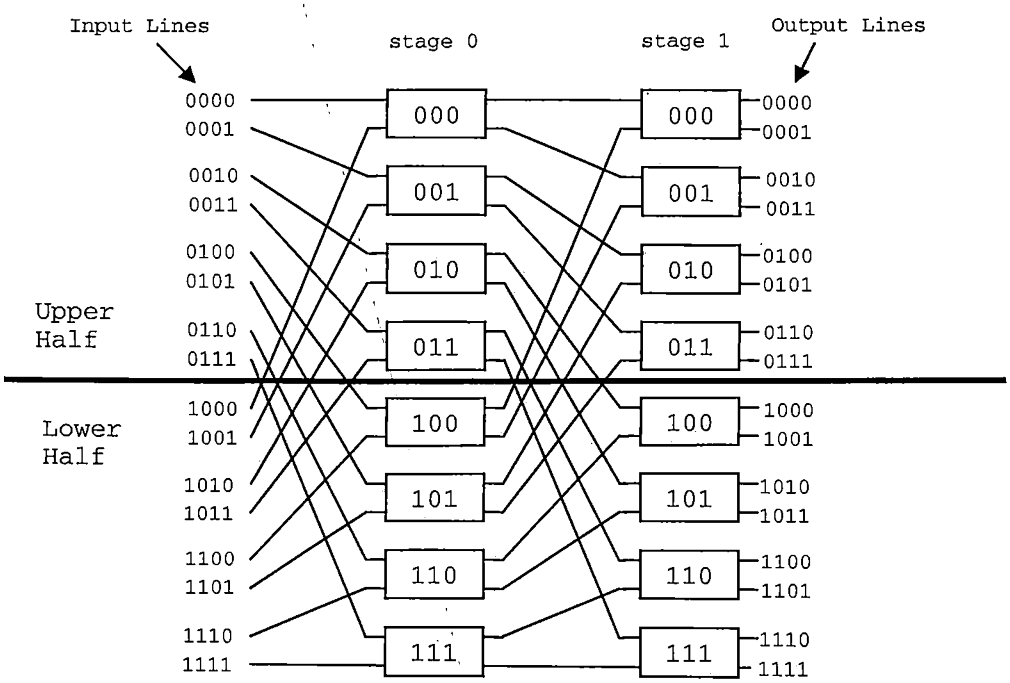


Figure 7. Shuffle-Exchange Switch Connections

Line Numbers. The line numbers are the inputs to each binary switch. It is important to define the switch numbers properly because the line numbers are based on the switch numbers. For $N=2^n$ input lines, the switches are numbered zero through $(2^{n-1} - 1)$. The input lines have (n) bits and the switch numbers have (n-1) bits. The input line to each switch is the (n-1) bit switch number with a "0" or "1" appended to the beginning. See Figure 7.

Baseline

The stages in the BL network are recursively divided in half as shown in Figure 8. For all $N=2^n$ inputs, there will be (n) stages with the stages getting subdivided n-1 times. The output lines from each switch connect to both a switch located in the upper half and a switch located in the lower half.

The upper half of each subdivision receives inputs from the upper output lines of the switches in the previous stage. The lower half of each subdivision receives inputs from the lower output lines of the switches in the previous stage.

Cube-Connected

The stages in the CC network are recursively divided in half as shown in Figure 9. For all $N=2^n$ inputs, there

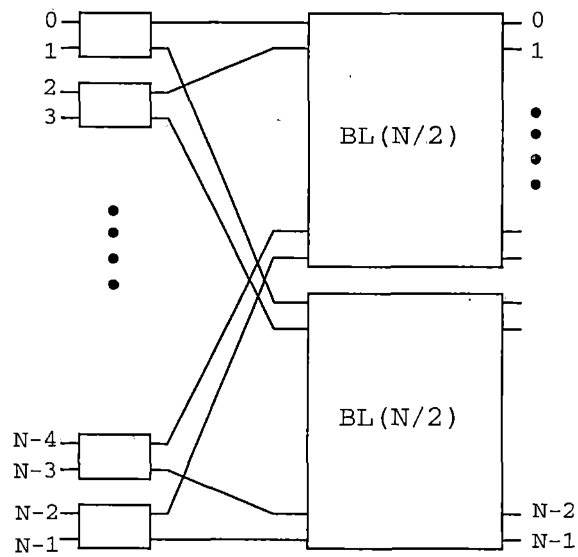


Figure 8. Baseline(N)

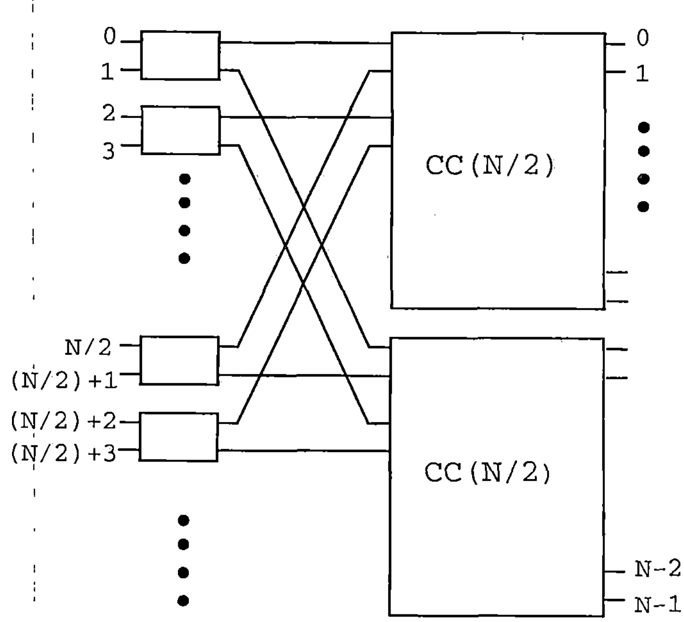


Figure 9. Cube-Connected(N)

will be (n) stages with the stages getting subdivided n-1 times. The output lines from each switch connect to both a switch located in the upper half and a switch located in the lower half.

To determine the switch connections from stage 0 to stage (n-1), one output line connects to the same numbered switch as depicted in Figure 10. If this output line is from a switch in the upper half, the upper output line connects to the upper input line of same numbered switch. If the output line is from a switch in the lower half, the lower output line connects to the lower input line of same numbered switch. To determine the connection for the second output line, the stage is used to define the bit position. The second output line connects to the switch different in only that bit position. The bit positions are counted from left to right with the MSB numbered as "0" and controlling stage 0 connections to stage 1. Proceeding from left to right, the LSB controls the connection from stage n-2 to stage n-1. The connection for the second output line will be from the lower output to the upper input or vice versa.

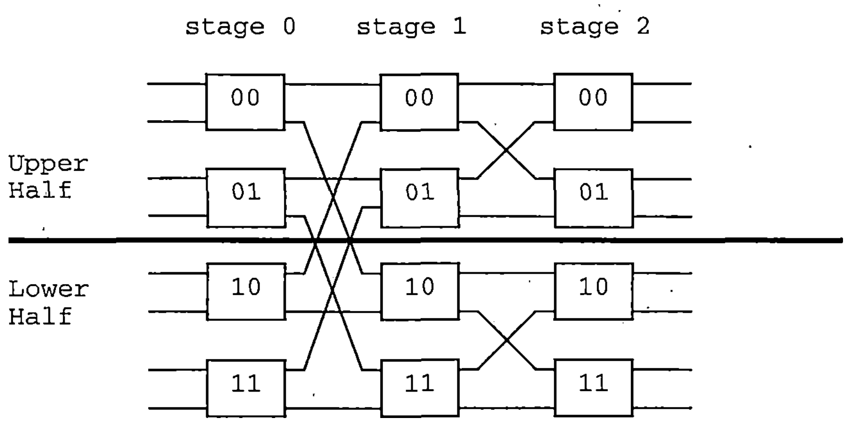


Figure 10. Cube-Connected Switch Connections

The upper half of each subdivision receives inputs from the upper output lines of the switches in the previous stage. The lower half of each subdivision receives inputs from the lower output lines of the switches in the previous stage.

Topological Equivalence

The SE network, BL network and CC network are all topologically equivalent when they have the same number of ($N=2^n$) input lines and the same number of (n) stages. There are a total of 2^{n-1} (or $N/2$) switches in each stage. Figure 11 depicts topologically equivalent networks. A method for demonstrating topological equivalence between interconnection networks is to transform one network to another by reordering the switches within each stage.

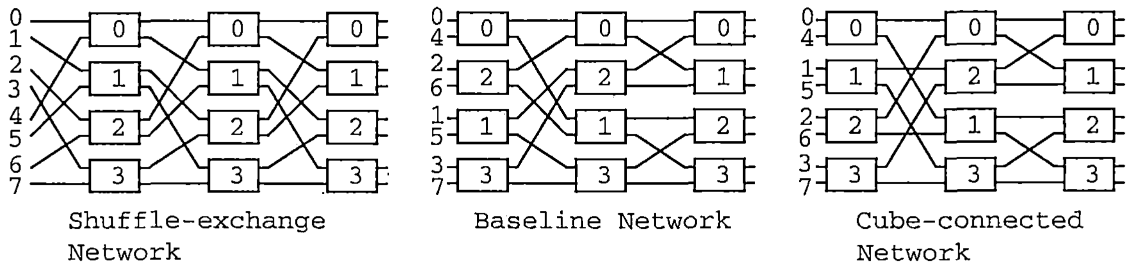


Figure 11. Topologically Equivalent Networks

Shuffle Exchange Reorganized as a Baseline Network

The number of stages is (n) while the final stage is numbered (n-1) because the stages are numbered starting with zero. In order to set the switch numbers for the SE network reorganized as a BL network, the initial stage must be set as defined in the next paragraph. The subsequent stages are set based on the switch settings in the previous stage. For example, if stage "0" is the initial stage, stage "1" is set based on stage "0" switch numbers. Stage "2" switch numbers are then determined based on stage "1" switch numbers, and so on.

Setting the Initial Stage. The SE initial stage switch numbers are numbered zero through (2^{n-1} -1) in ascending order. The number of bits is (n-1). The reorganized BL network initial stage switch numbers are also numbered zero through (2^{n-1} -1) in ascending order except the numbers are incremented starting with the MSB instead of the LSB. In ordinary binary notation, the pattern for 8 switches is 000, 001, 010, 011, 100, 101, 110, 111. To count in reverse, the pattern is 000, 100, 010, 110, 001, 101, 011, 111.

Setting Subsequent Stages. Once the initial stage is set, the subsequent stage switch numbers are set from the switch numbers in the same position in the previous stage. A left-circular shift (LCS) is performed on the previous stage switch number to determine the switch number in the next stage. The stage number dictates how many bits on which the LCS is performed. The initial stage is defined as stage "0". Therefore, the next stage is stage "1". A LCS is performed on one bit, namely the least significant bit (LSB), of the switch number. A LCS on only the LSB yields the same number. This explains why in going from stage 0 to stage 1, the switch numbers remain the same. When defining stage "2" switch numbers, a LCS is performed on the "2" LSBs of the switch numbers in stage "1". For example, "110" becomes "101" in stage 2 and "011" becomes "011" in stage 2. One continues in this manner until stage number (n-1) is defined; where all switches are numbered zero through (2^{n-1} -1) in ascending order. Figure 12 is an example of a 16-input SE network reorganized as a BL network.

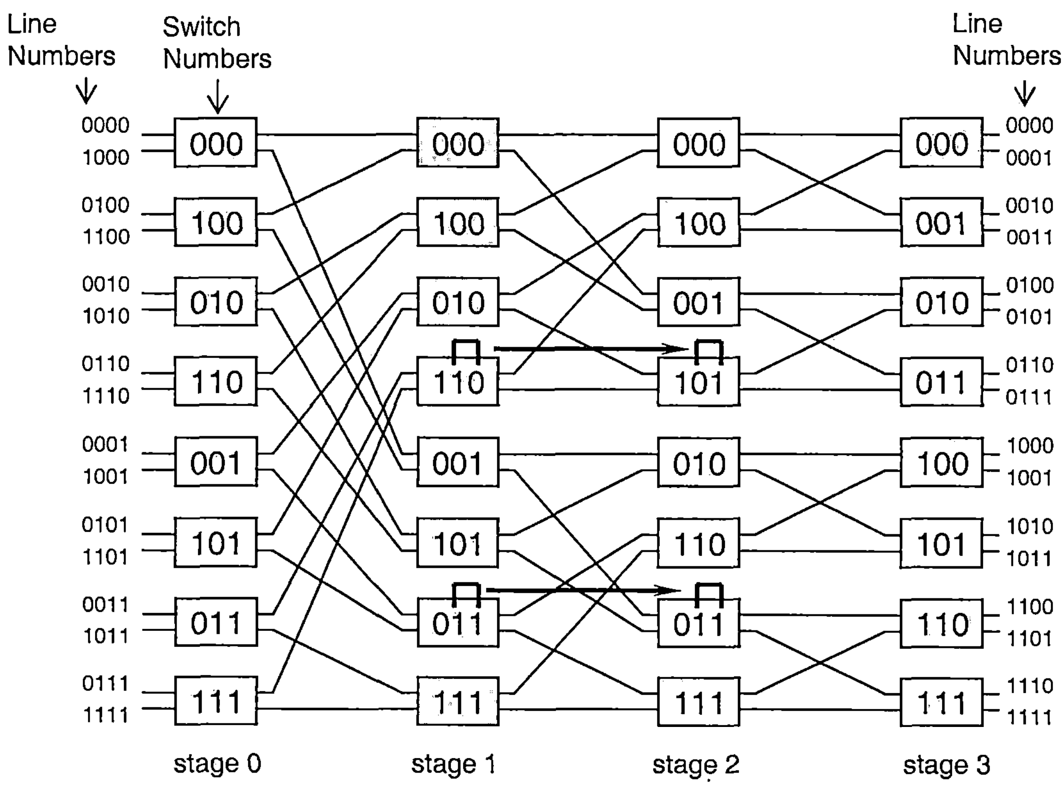


Figure 12. 16-Input, Shuffle-Exchange Network Reorganized as a Baseline Network

Shuffle Exchange Reorganized as a Cube-Connected Network

Setting the Initial Stage. The reorganized CC network initial stage switch numbers are set as numbers zero thru (2^{n-1} -1) in ascending order. This switch numbering is the same as the switch numbering for the initial stage of the SE network.

Setting Subsequent Stages. Once the initial stage is set, the subsequent stage switch numbers are set from the switch numbers in the same position in the previous stage. A LCS is performed on the previous stage switch number to determine the switch number in the next stage. The number of bits is (n-1). The LCS is performed on all (n-1) bits. This methodology for setting subsequent stages is used in setting all stages from stage "1" to stage number "n-1", which is the final stage.

Reorganized Baseline.Cube-Connected Network

A 16-input SE network is shown in Figure 13. An N-input BL network with (n) stages concatenated with an N-input CC network with (n) stages forms a composite BL.CC network for $N=2^n$ input lines. The last stage of the BL network is the same as the first stage of the CC network as shown in Figure 14. When this redundant stage is combined, the composite BL.CC network has (2n-1) stages. Figure 15 shows a 7-stage, 16-input SE network reorganized as a composite BL.CC network. The first (n-1) stages are the BL network and the next (n) stages represent the CC network.

The stage interconnections in the SE network are reproduced in the composite BL.CC network. For example, switch "0" is connected to switches "0" and "1" in every stage in both the SE network and the BL.CC network.

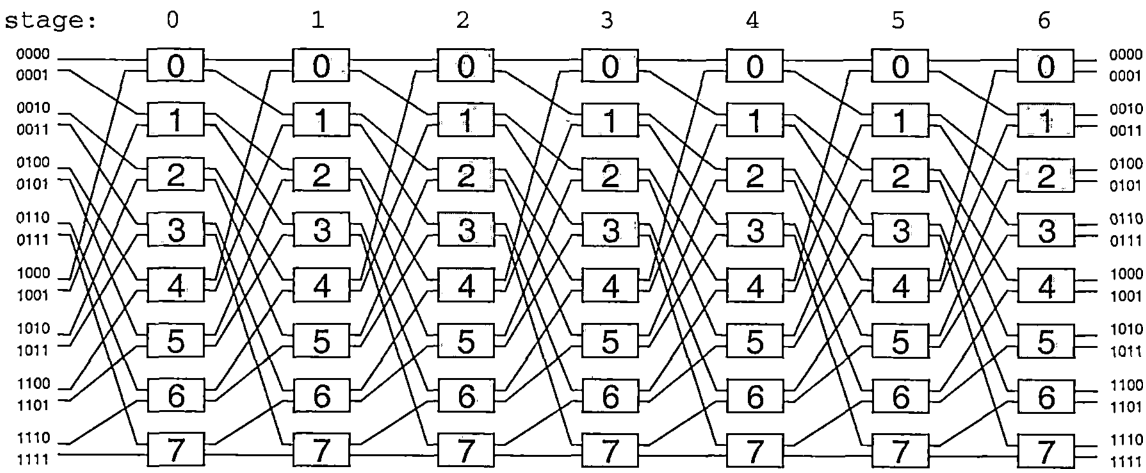


Figure 13. 16-Input, 7 Stage Shuffle-Exchange Network

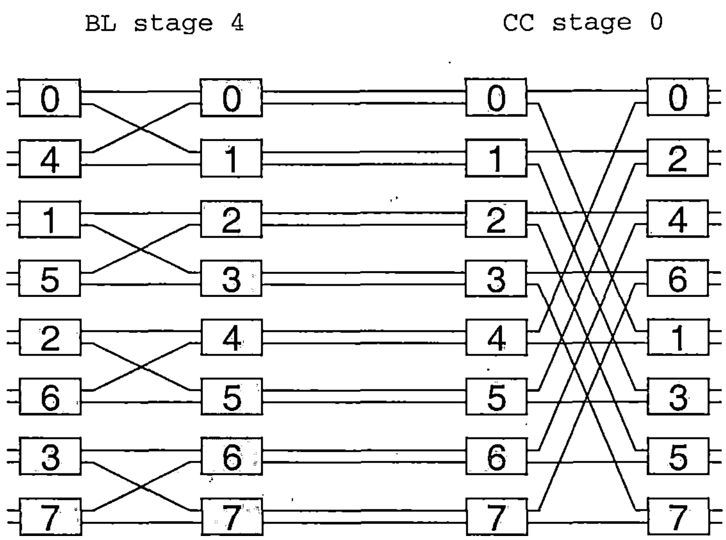


Figure 14. Redundant Stage of Baseline.Cube-Connected Network

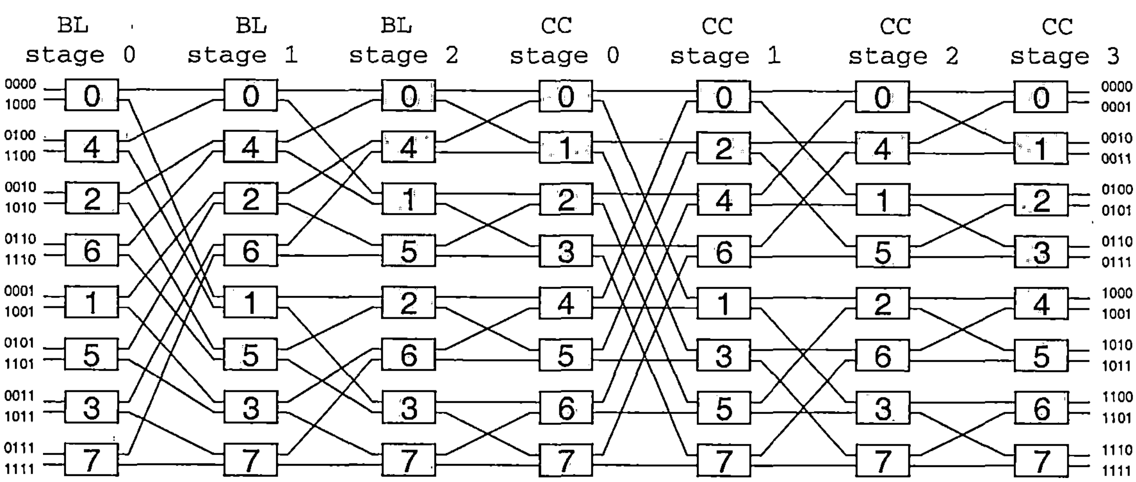


Figure 15. 16-Input, 7 Stage Baseline.Cube-Connected Network

Functional Equivalence

Two interconnection networks are functionally equivalent if they realize the same set of permutations. When two interconnection networks are topologically equivalent, their functional equivalence can be established by relabeling their inputs [11][12].

Topological equivalence between the SE and BL.CC network has been demonstrated. Functional equivalence can be shown by renaming the inputs to the BL.CC network. The renaming of the input lines to the BL.CC network, as depicted in Figure 15, simulates the SE network.

The output produced by (2n-1) stages of the SE network and (2n-1) stages of the BL.CC network is the same for any given permutation. Both networks relate the output to the input in the same way and therefore realize the same permutations. This establishes that the N-input BL.CC network is functionally equivalent to the N-input SE network, when $N=2^n$.

CHAPTER FOUR

CONTROL HEURISTIC DEVELOPMENT

Design

The approach to establishing the heuristic on the rearrangeability of (2n-1) stages of the SE network involves developing a heuristic on the rearrangeablilty of (2n-1) stages of the composite BL.CC network. The control heuristic uses fundamental principals of the BL and CC networks. The first (n-1) stages are the BL portion of the composite BL.CC network. The permutation is routed through the first (n-1) stages such that at stage number (n-1) the permutation is reordered as a CC permutation. Once this is achieved, the well-known bit-matching algorithm for routing TAs through a CC network is used to complete the routing of the inputs through the rest of the network.

Conditions and Requirements

Cube-Connected Network Routing

Cube-Connected Permutation Requirements. The algorithm for routing inputs through a CC network is documented and well understood. A cube-connected permutation (CCP) is necessary for routing inputs through a

CC network. A CCP has the following requirements for 16 inputs:

- All combinations of the most significant bit (MSB) on each switch (this is simply 0 and 1). For example, switch 0 must have 0xxx and 1xxx, where x could be 0 or 1.
- All permutations of the 2 MSB on all switches equivalent modulo 2^{n-2}. This is the same as stating all switches different in only their MSB have all permutations of the 2 MSBs. For example, switches 0 and 4 must have 00xx, 01xx, 10xx, and 11xx.
- All permutations of the 3 MSB on switches equivalent modulo 2^{n-3}. This is the same as stating all even switches (and all odd switches) have all permutations of the 3 MSBs. For example, switches 0, 2, 4, and 6 must have 000x, 001x, 010x, 011x, 100x, 101x, 110x, and 111x.

Figure 16 shows three examples of CCPs and the attributes for 16 inputs.

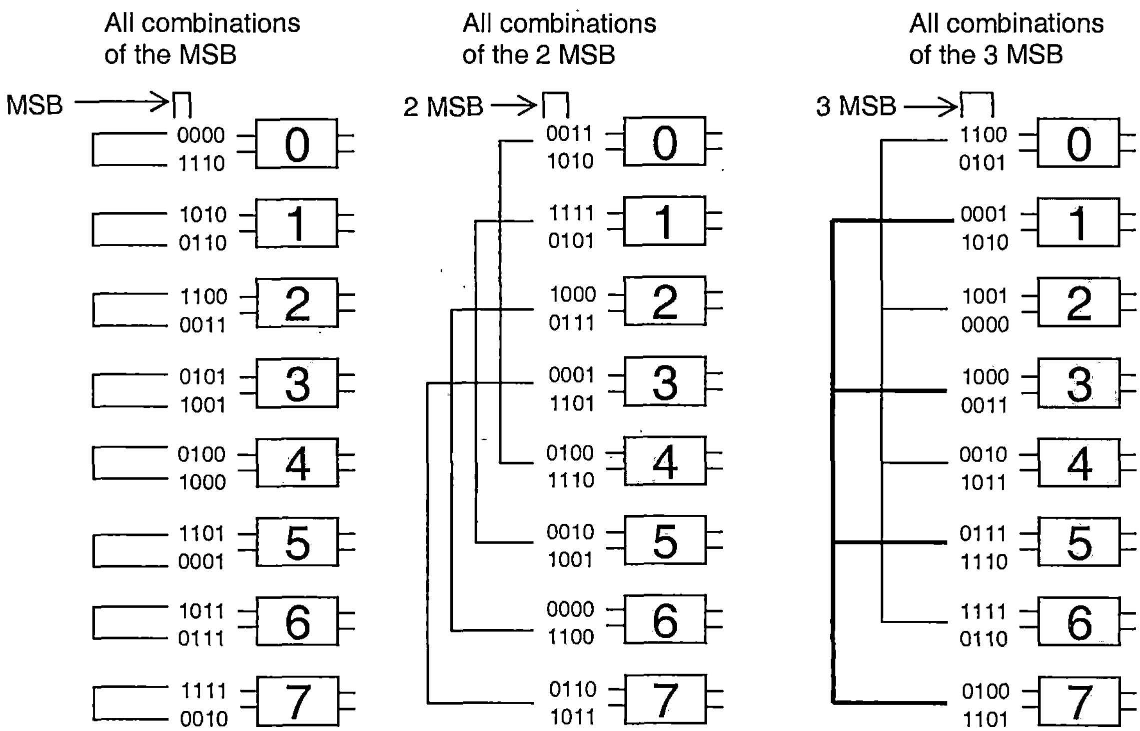


Figure 16. Cube-Connected Permutations

<u>Routing Algorithm</u>. The i^{th} bit controls the setting for the switch in stage n-i-1. The initial stage is stage "0". Figure 17 depicts bit numbering for 16 inputs with n=4. For example, in setting stage "0", the MSB, bit number 3, determines the switch setting.

x x x x
3 2 1 0

Figure 17. 16-Input Bit Numbering

Inputs are routed once the control bit is determined. If a "1" is on the upper input line in the control bit position, the switch setting is cross. If a "1" is on the lower input line, the switch setting is through. The CC control scheme is depicted in Figure 18.

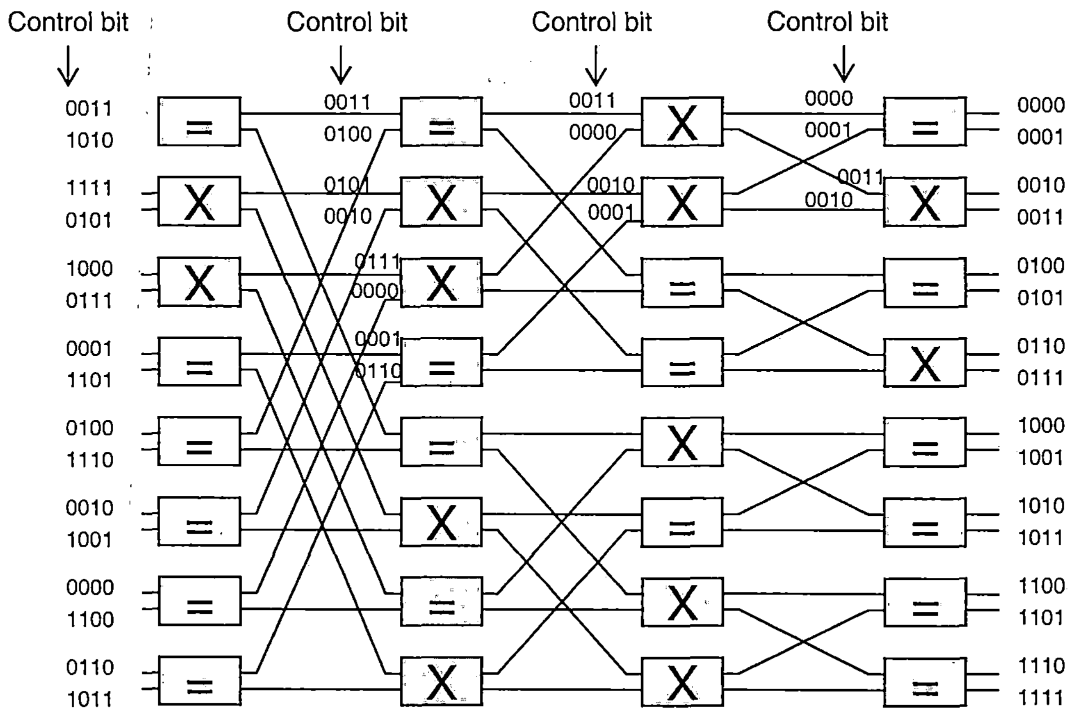


Figure 18. Cube-Connected Network Control Algorithm

Baseline Network Routing

Purpose. The majority of work done for this thesis was in developing the conditions for passing permutations through the BL segment of the BL.CC network. The purpose of routing the inputs through the BL segment is to realize a CCP in BL stage (n-1), which is also CC stage 0.

Routing Conditions. Two conditions (conditions 1 and 2) were discovered, which must be adhered to in order to realize a CCP in CC stage 0. Conditions 1 and 2 are defined for N=16 (n=4). The conditions operate on the TAs as they are routed through the BL portion of the network. The TAs, shown in Figure 19, are a random input permutation.

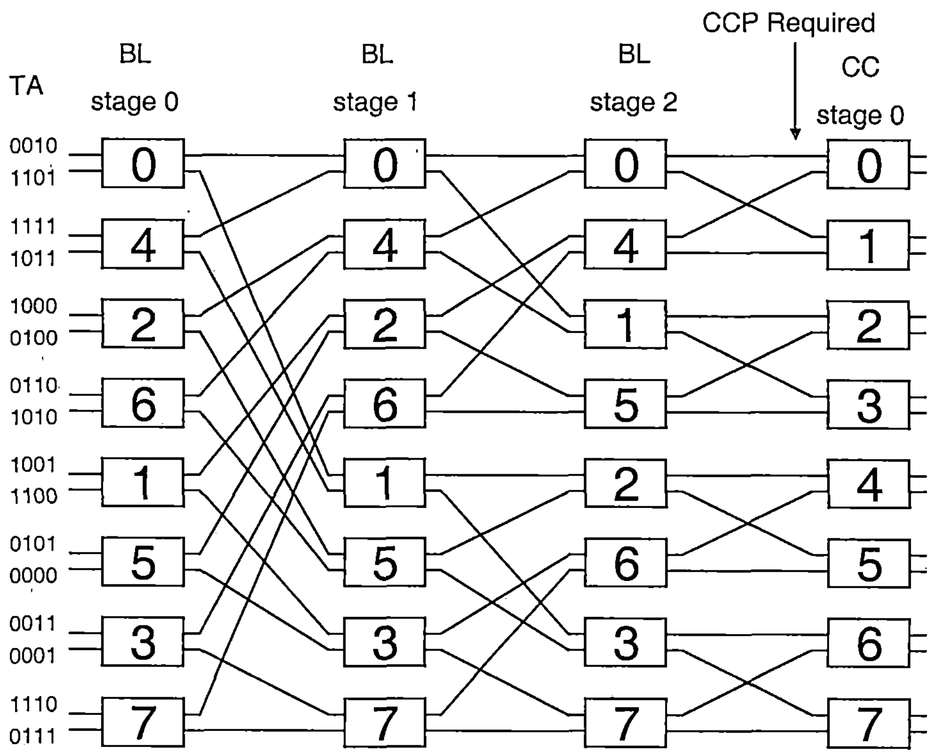


Figure 19. Baseline Portion of Baseline.Cube-Connected Network

* Condition 1: In every stage, there must be an equal number of zeros and ones in the MSB position within each subdivision. That is, going from stage 0 to stage 1, inputs to switches 0, 4, 2, and 6 must have four 0xxx and four 1xxx. Going from stage 1 to stage 2, switches 0 and 4 must have two 0xxx and two 1xxx. Figure 20 depicts the subdivisions.

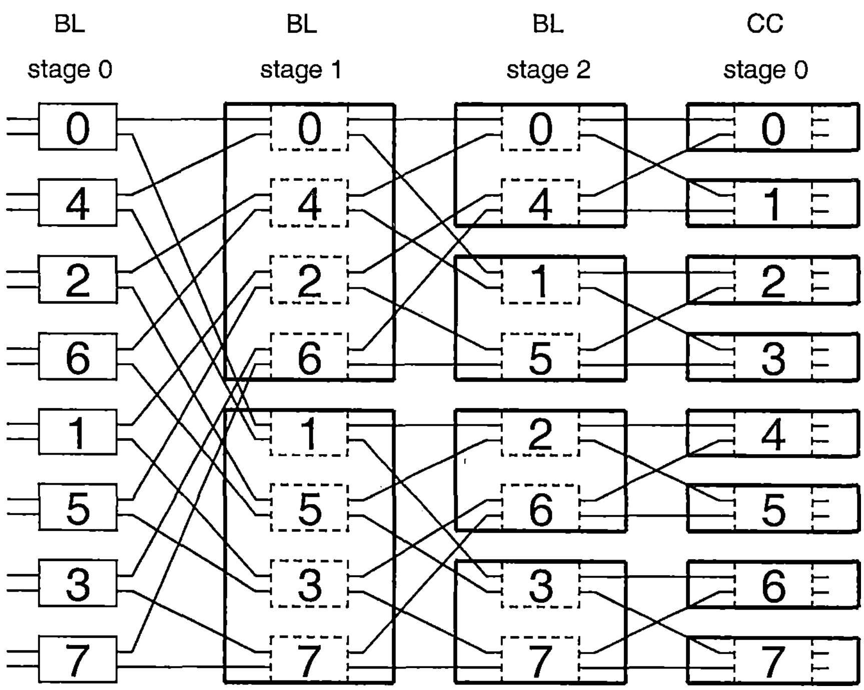


Figure 20. Baseline Stage Subdivisions

* Condition 2: In BL stage 2, there must be a pair of the 2 MSBs on switches equivalent modulo 2. That is, switches 0, 4, 2, and 6 in stage 2 must have two

00xx, two 01xx, two 10xx, and two 11xx. See Figure 21. Condition 2 must be adhered to in conjunction with the constraints defined in the upcoming paragraph titled "Condition 2 Locked Pair Constraints." The condition 2 constraints exist because of the CCP requirements. The CCP requirements are defined and then the condition 2 constraints are explained.

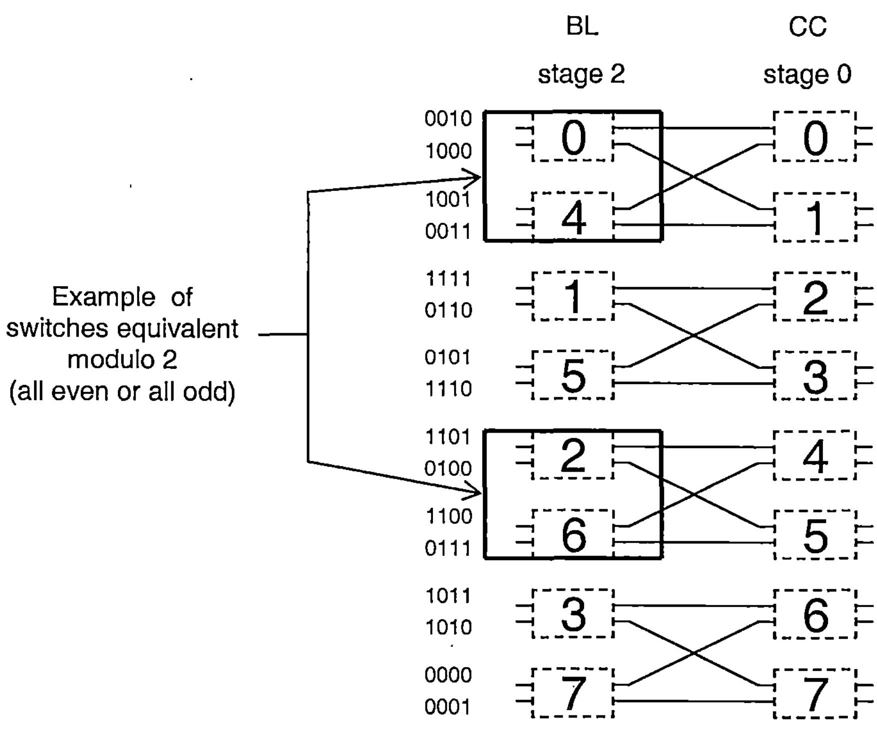


Figure 21. Baseline Stage Condition 2 Criteria

Routing Requirements. It is necessary to follow conditions 1 and 2 to set the switches in BL stages 0 and 1 because these conditions control how the TAs will be arranged as inputs to BL stage 2. It is necessary to follow the CCP requirements to set the switches in BL stage 2. The CCP requirements are enforced so that TAs are arranged as a CCP for input to CC stage 0. The requirements apply to CC stage 0.

- MSB Requirement: All switches have all permutations of the MSB.
- 2 MSB Requirement: All switches equivalent modulo 4 have all permutations of the 2 MSBs.
- 3 MSB Requirement: All switches equivalent modulo 2 have all permutations of the 3 MSBs.

Routing Constraints. Conditions 1 and 2 are necessary but not sufficient to guarantee that the inputs to BL stage 2 can be arranged such that a CCP can be generated in CC stage 0. When the number of constraints exceeds the number of switches that can be freely set, conflicts occur. When a constraint exists between switches, one switch automatically sets the other. The goal is to reduce, minimize, and even eliminate constraints so that a CCP can be generated.

Condition 2 Locked Pair Constraints. It was discovered that constraints exist for the pair of 2 MSBs on switches equivalent modulo 2 required by condition 2. These constraints result from the necessary requirements defined for having a CCP in CC stage 0. To reiterate the CCP requirements for N=16 inputs: all permutations of the MSB on each switch, all permutations of the 2 MSBs on switches equivalent modulo 4, and all permutations of the 3 MSBs on switches equivalent modulo 2. A "locked pair" is a pair of TAs that always exists together on a switch in CC stage 0. Following is an explanation of how the CCP requirement for the MSB creates locked pairs and how other constraints follow when enforcing the CCP requirement for the 2 MSBs and the 3 MSBs in the presence of locked pairs.

Locked Pair Creation. Within each subdivision of BL stage 2, there exists a set of switch numbers differing in only their MSB, refer to Figure 22. The switches within each subdivision are defined as partners P1 and P2. Because there is a requirement for an even distribution of the MSB within each subdivision, if the MSB differs on P1, this automatically implies a difference on P2. When this occurs, one switch setting automatically sets the other switch. A constraint called "locked pairs" is created because the CCP requirement for the MSB requires all permutations of the MSB on each switch in CC stage 0 and the pairs that exist on the switches in CC stage 0 are "locked" together. When a "0xxx" is sent to an upper switch by P1, a "1xxx" must be sent to the upper switch by P2, and vice versa. P2 has no freedom in choosing its switch setting. This leads to the corollary that the worst case in BL stage 2 is when each switch has all permutations of the MSB (i.e., every switch has a TA with a MSB zero and another with a MSB one). Worst case is defined as the case when there is reduced freedom in setting switches because one switch setting automatically sets the other.

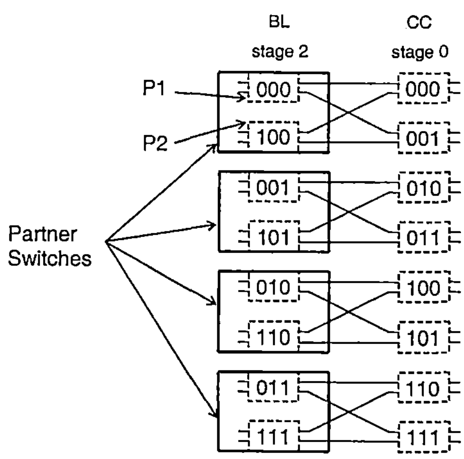


Figure 22. Partner Switches

Locked Pairs and the Cube-connected Permutation Requirement for 2 MSBs. The even switches in BL stage 2, highlighted in Figure 21, produce the inputs for switches 0&4 and switches 1&5 in CC stage 0. The even switches in BL stage 2 are required to have a pair of the 2 MSBs so that in CC stage 0 the switches equivalent modulo 4 can have all permutations of the 2 MSBs. When the even switches have a pair of the 2 MSBs, the odd switches meet this criterion by default and switches 2&6 and switches 3&7 can have all permutations of the 2 MSBs. If locked pairs exist on any one set of partner switches, there are more constraints in setting the switches to meet the CCP requirement for the 2 MSBs because there is less flexibility in setting the BL stage 2 switches. If locked pairs exist on both sets of partner switches (i.e., on all the even numbered switches or all the odd numbered switches), flexibility in setting the BL stage 2 switches such that a CCP is generated in CC stage 0 is reduced even more.

Locked Pair Conflict. An example of a conflict is shown in Figure 23. The locked pair "0000" and "1001" exists on switch 0 in CC stage 0. The pair "1000" and "0011" exists on switch 6 in BL stage 2. No matter how switch 2 is set in BL stage 2, one of the outputs must be an input to switch 4. As such, switches 0&4 can never have all permutations of the 2 MSBs. Changing the switch setting for switch 0 in BL stage 2 to "cross" means that switches 1&5 can never have all permutations of the 2 MSBs. Under no circumstances will this permutation create a CCP in CC stage 0.. The CCP requirement for the 2 MSBs will always be violated. The conditions stated for the even switches apply likewise to the odd switches with the rule that the odd switches produce inputs for switches 2&6 and switches 3&7 in CC stage 0.

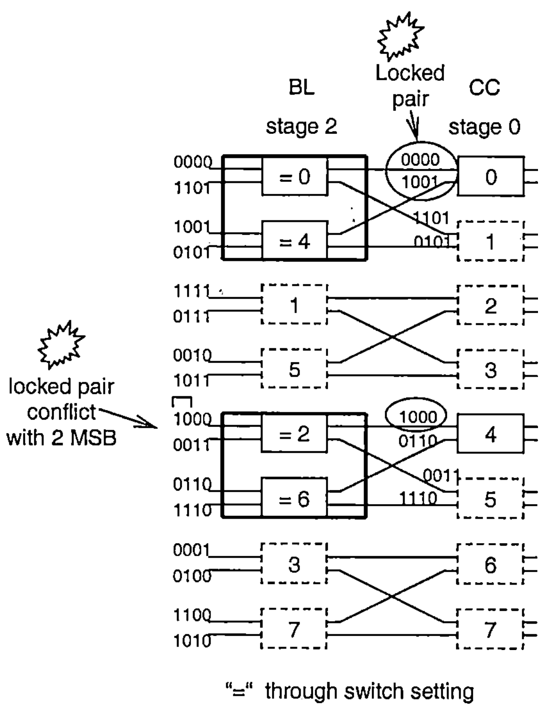


Figure 23. 2 Most Significant Bits Locked Pair Constraint

Locked Pairs and the Cube-connected Permutation Requirement for 3 MSBs. The locked pairs created by the partner switches in BL stage 2 also create a constraint between the 3 MSBs in CC stage 0 and increase the chance of a conflict. If a locked pair exists in CC stage 0 and does not create a 2 MSB conflict, the 3 MSBs must be examined. The 3 MSBs of that locked pair cannot exist together on the same switch in BL stage 2. This constraint exists because every switch in BL stage 2 produces an input for an even switch and an input for an odd switch. If the "locked pair" 3 MSBs exist together on a switch in BL stage 2, at least one of the 3 MSBs will be repeated on an even switch or on an odd switch, depending on how the switch is set. As the example in Figure 24 shows, the 3 MSB "001" on switch 7 in CC stage 0 conflicts with the 3 MSB "001" on switch 1 in CC stage 0. This conflict means that a CCP cannot be achieved in CC stage 0 because the CCP requirement for the 3 MSBs requires that all permutations of the 3 MSBs exist on both the even and the odd switches. This is an example of a permutation that satisfies the CCP requirement for the 2 MSBs but fails to satisfy the CCP requirement for the 3 MSBs.

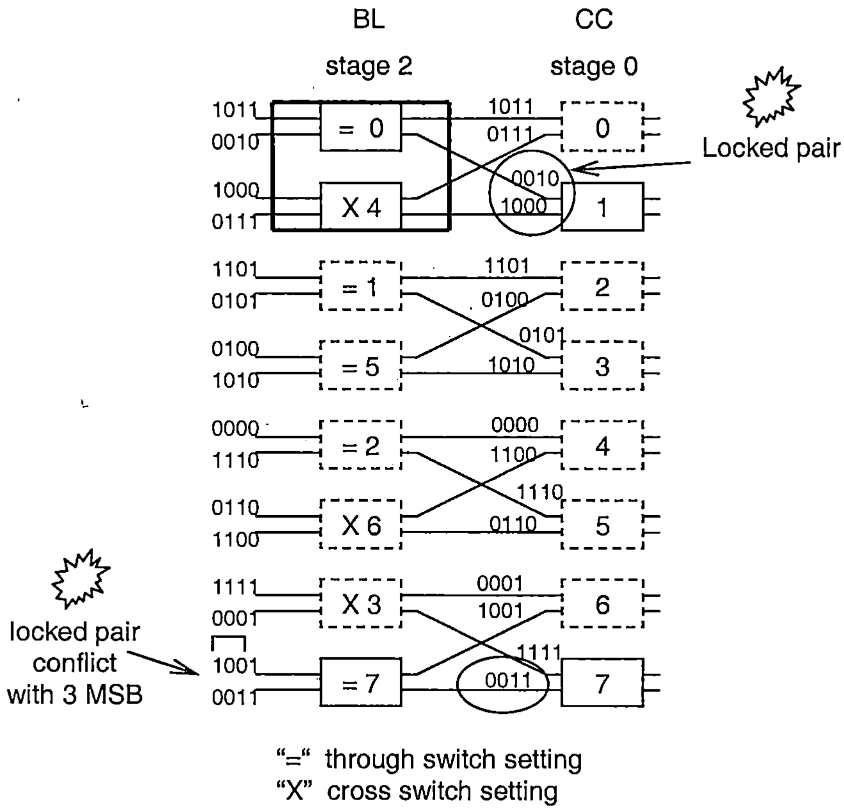


Figure 24. 3 Most Significant Bits Locked Pair Constraint

CHAPTER FIVE

CONTROL HEURISTIC IMPLEMENTATION

Methodology

Introduction

A control heuristic was implemented to pass N=16 inputs through the BL portion of the BL.CC network and realize a CCP in CC stage 0. When N=16, there are 16! (approximately 20.9 trillion) permutations. This is called a heuristic because the conditions, which control the heuristic, are necessary but not sufficient to generate a CCP for all 16! permutations. This chapter outlines the logic for setting the BL switches. The heuristic is given in Appendix A.

Approach

Switch Setting Scheme. The default switch setting is "through" for all switches in every stage. The switches are set in ascending order from position one to position N/2. See Figure 25. After each switch is set, a check is made to determine if a condition or requirement is violated or a conflict is detected. If there is a violation or conflict, the switch is reset to "cross".

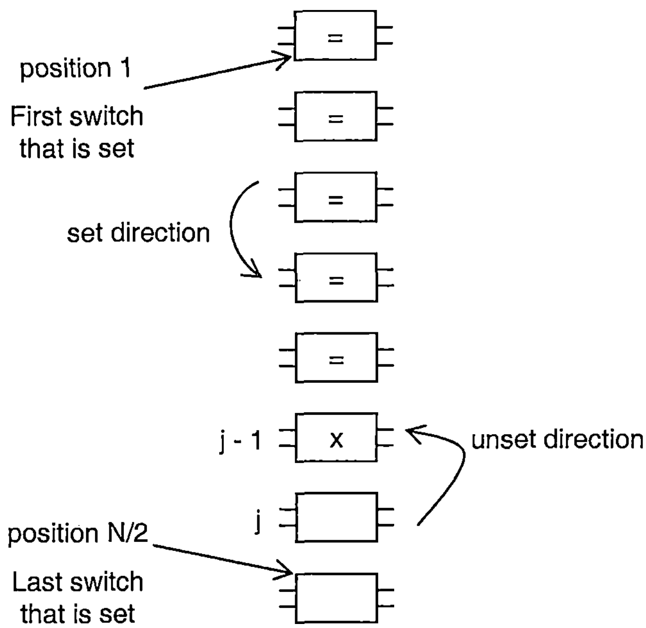


Figure 25. Baseline Stage Switch Setting Scheme

Switch Reset Scheme. If resetting the switch at position j to "cross" does not satisfy the current requirements, the switch at position j is unset and the switch at position j - 1 is reset. The switches within a stage are unset in the reverse order in which they are set so that the previous switch can be reset, as shown in Figure 25. The switches are unset in reverse order until an acceptable switch setting is found. When an acceptable switch setting is found, the switch setting scheme proceeds forward. The stage switch setting scheme ends when an acceptable setting is found for all N/2 switches or the switch in position 1 has been reset and the current requirements still cannot be satisfied.

Stage Termination. If a condition or requirement cannot be satisfied or a conflict cannot be resolved by resetting the switches within a stage, a message is sent that the switches for that stage cannot be set; and, the control heuristic terminates. Backtracking from one stage to a previous stage is not allowed to minimize the time complexity of the program.

Condition 1 Implementation

Condition 1 is implemented when setting the BL switches as follows. Each subdivision of each stage, as shown in Figure 26, is forced to have an equal number of zeros and ones in the MSB position. This guarantees that in BL stage 2 there will not be a problem setting the switches to distribute a zero and a one to each switch in CC stage 0. Since each stage is recursively divided, this rule must be enforced. If it is not enforced, there is a guarantee that there will be a violation of the CCP requirement for the MSB. The switch interconnections have been removed for clarity.

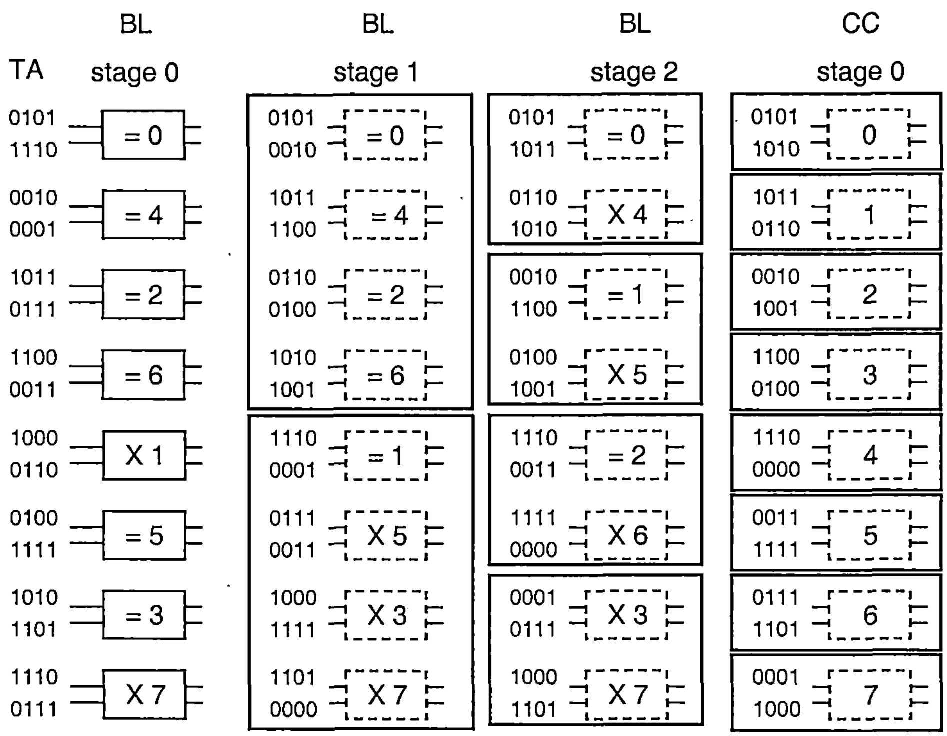


Figure 26. Condition 1 Example

Baseline Stage 0. In setting the switches in BL stage 0 to produce the inputs for BL stage 1, condition 1 is enforced. The MSB ones and zeros are evenly distributed at the end of setting the stage. The default switch setting is "through". The switch setting "cross" is used when resetting within the stage is required to balance the distribution of zeros and ones. Condition 1 can always be met when setting BL stage 0 switches.

Baseline Stages 1 and 2. Condition 1 is not explicitly implemented when setting the switches in BL stages 1 and 2. Condition 1 defaults to true when condition 2 and the CCP requirements are satisfied. For example, there cannot be a pair of the 2 MSBs on the even switches without there being an equal number of zeros and ones in the MSB position, refer to Figure 26.

Condition 2 Implementation

Condition 2 is implemented when setting BL stage 1 switches. The switches are set to produce the inputs for BL stage 2 and ensure that there are all permutations of the 2 MSBs on switches equivalent modulo 2. If condition 2 cannot be enforced after resetting the BL stage 1 switches, a message is sent that the BL stage 1 switches cannot be set; and, the control heuristic terminates.

Condition 2 Locked Pair
Constraint Implementation

Baseline Stage 1, 2 MSBs. The condition 2 locked pair constraint is checked after all BL stage 1 switches have been set to meet condition 2. If the constraint exists and causes a conflict with the CCP requirement for the 2 MSBs, the switches are reset. If resetting does not allow condition 2 to be met while also eliminating conflicts, a message is sent that the BL stage 1 switches cannot be set; and, the control heuristic terminates.

Baseline Stage 1, 3 MSBs. The condition 2 locked pair constraint is checked for the 3 MSBs after the constraint is checked for the 2 MSBs. If the condition 2 locked pair constraint exists and causes a conflict with the CCP requirement for the 3 MSBs, the switches are reset. If resetting does not remove the conflict, an error message is sent that the BL stage 1 switches cannot be set; and, the control heuristic terminates.

Cube-Connected Permutation Requirements Implementation

Baseline Stage 2, MSB. The CCP requirement for the MSB is enforced when setting the BL stage 2 switches. The BL stage 2 switches can always be set to meet the CCP MSB requirement when BL stage 1 is successfully set.

Baseline Stage 2, 2 MSBs. The CCP requirement for the 2 MSBs is enforced when setting the BL stage 2 switches. This requirement is enforced in conjunction with the CCP requirement for the MSB. The BL stage 2 switches can always be set to meet the CCP 2 MSB requirement when BL stage 1 is successfully set and the 2 MSB locked pair conflict has been avoided.

Baseline Stage 2, 3 MSBs. The CCP requirement for the 3 MSBs is enforced when setting the BL stage 2 switches.

This heuristic detects and tries to avoid conflicts created by locked pairs for the 3 MSBs. If the BL stage 1 switches have been set, then any locked pair constraints for the 3 MSBs will not cause a failure in setting BL stage 2 switches. The BL stage 2 switches cannot always be set to meet the CCP 3 MSB requirement. This is because the BL stage 2 switches cannot be set to satisfy the CCP 3 MSB requirement without also being set to satisfy both the CCP 2 MSB requirement and the CCP MSB requirement. Figure 27 is an example of this type of conflict.

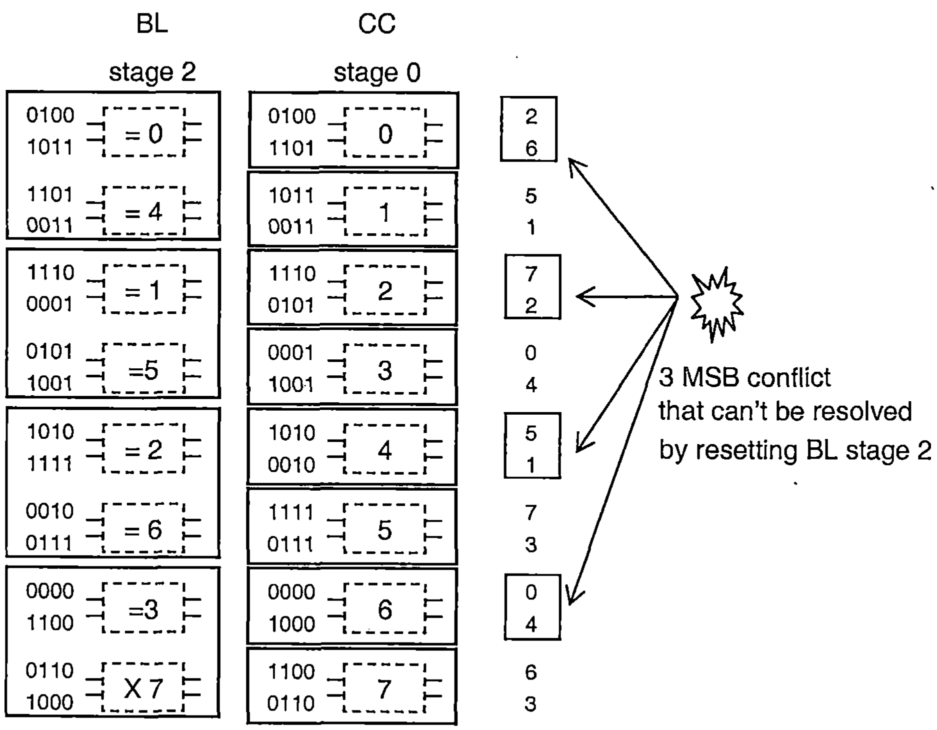


Figure 27. Baseline Stage 2 Conflict

Interdependencies Among Requirements

Failure in setting the BL stage 2 switches for the 3 MSB CCP requirement occurs because this requirement is enforced in conjunction with the CCP requirement for the 2 MSBs, which is enforced in conjunction with the CCP requirement for the MSB. The interdependencies and constraints caused by satisfying all three CCP requirements at the same time create scenarios where there is no acceptable switch setting for permutations as they are arranged in BL stage 2. Interdependencies result when one switch sets another. The BL stage 2 switches cannot always be set to meet all of the 3 CCP requirements at the same time. When resetting within BL stage 2 fails to produce a CCP in CC stage 0, an error message is sent that the BL stage 2 switches cannot be set; and, the control heuristic terminates.

Results

Scope

The model chosen for this control heuristic is a 16-input model. Random permutations of numbers 0 through 15 are generated as input for the BL control heuristic. The control heuristic sets the switches in the BL portion of the BL.CC network. The algorithm for setting the switches in the CC portion of the BL.CC network always works as long as a CCP is generated as input for CC stage 0; therefore, it is not necessary to display results for the CC portion.

Type of Results

The control heuristic generates results for BL stages 0 through 2 and CC stage 0. There are three types of results. The first type, category A, demonstrates that the control heuristic can set the switches and generate a CCP in CC stage 0. The second type, category B, demonstrates that the control heuristic can fail to set the switches in BL stage 2 and therefore a CCP is not generated in CC stage 0. The third type, category C, shows that the control heuristic can fail to set the switches in BL stage 1 and therefore a CCP is not generated in CC stage 0. Examples of the three types of results are shown in Appendix B. Although there are three types of results, the control heuristic either succeeds in generating a CCP in stage 0 or it fails. Category A results are successful. Category B and C results are failures.

Input Permutations

The question to be answered is how many of the possible 16! permutations this control heuristic succeeds in passing through the BL portion of the BL.CC network. In order to avoid exhaustively running all 16! permutations, a scheme is used that generates a uniform distribution of random permutations [13]. The algorithm is depicted in Figure 28. Of the 16! possible permutations, each receives an equal probability of being generated.

```
N = 16;
for j = 1 to x {
   for i = 0 to N-1, do a[i] = i;
   for i = 0 to N-2, do swap(a[i], a[Random(i,N-1)]);
}
```

Figure 28. Random Permutation Algorithm

Two loops are implemented to produce "x" number of random permutations, and "x" is a minimum of 10 and a maximum of 100 million. The "x" number of random permutations is considered a set. Random permutations were run in different size sets to discover patterns and note anomalies in the results. The results shown in Table 1 are based on using this uniform distribution of random permutations as inputs to BL stage 0 and routing the permutations as TAs through the BL portion of the BL.CC network.

Table 1. Control Heuristic Results

Size of set x	**Number of Successes**	**Number of Failures**	**Running Time**
10a	8	2	0.01 secs
10b	5	5	0.01 secs
10c	2	8	0.01 secs
100a	54	46	0.09 secs
100b	57	43	0.08 secs
1000a	557	443	1.03 secs
1000b	562	438	1.06 secs
10,000a	5,661	4,339	10.22 secs
10,000b	5,577	4,423	10.39 secs
100,000a	56,111	43,889	102.11 secs
100,000b	56,194	43,806	102.94 secs
1,000,000a	560,923	439,077	~17 mins
1,000,000b	559,907	440,093	~17 mins
10,000,000a	5,606,712	4,393,288	~2.75 hrs
10,000,000b	5,608,384	4,391,616	~2.75 hrs
100,000,000a	56,072,347	43,927,653	~28 hrs
100,000,000b	56,095,683	43,904,317	~28 hrs

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS

Summary

When setting the switches in BL stages 0, 1, and 2, there is a known condition and requirement for the MSB. There must be an equal number of MSB zeros and ones within each subdivision, condition 1 and CCP MSB requirement. When setting the switches in BL stages 1 and 2, there is a known condition and requirement for the 2 MSBs, condition 2 and CCP 2 MSB requirement. When setting the switches in BL stage 2, there is a known requirement for the 3 MSBs, CCP 3 MSB requirement. Other constraints have been exploited when setting BL stage 2 switches for condition 2. When there are locked pairs, sometimes steps can be taken to avoid conflicts. Unfortunately, these conditions, requirements and conflict avoidance techniques are not enough to guarantee that a CCP will be generated in CC stage 0.

Conclusions

The goal was to discover a method to pass all permutations. The results show that when the conditions and requirements of this research are met, a CCP is generated approximately 56% of the time. The results are consistent and demonstrate that this control heuristic succeeds more than 50% of the time.

There is more than one way to set the BL switches to pass a permutation. This heuristic defines one way of setting the switches to achieve this.

This research serves as the foundation for defining the final algorithm that will allow all permutations to pass through the BL portion of the network and result in a CCP. Once all permutations can pass through the BL stages, all permutations will pass through the entire BL.CC network. The BL.CC network is functionally equivalent to the SE network and therefore can be used to prove that a 16-input SE network is rearrangeable.

Recommendations and Future Work

There are several areas that can be explored to increase the success of this control heuristic. Care should be taken to discover conditions that are both necessary and sufficient.

Case Studies

One has to avoid analyzing cases that sometimes fail and then removing all cases that fit that scenario. For example, when the MSB is different on the partner switches in BL stage 2, the chance of conflicts increases due to multiple constraints. However, there are cases when permutations successfully pass although this condition exists. See Figure 29. If the control heuristic is designed to remove this scenario, one has to make sure that probability of success increases and does not decrease.

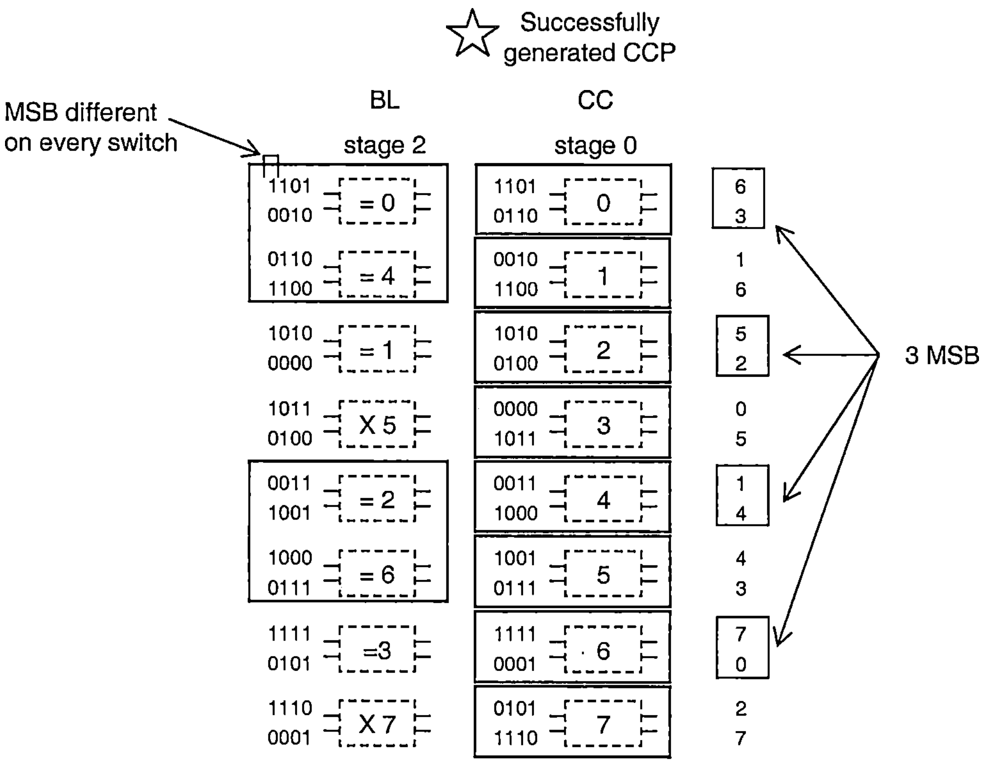


Figure 29. Successfully Generated Cube-Connected Permutation

Condition 3

A condition needs to be found to control the 3 MSBs in BL stage 0 or BL stage 1. This would eliminate some of the constraints encountered when setting the switches in BL stage 2. The necessity to satisfy all three CCP requirements at the same time causes conflicts that sometimes cannot be solved by resetting the BL stage 2 switches. Constraints should be removed earlier in the heuristic that reduce the chance of the 3 MSBs conflicting in setting BL stage 2.

3 MSBs Conflict Reduction

There are cases in which conflicts between the 3 MSBs in BL stage 2 can be reduced. If the 3 MSBs are evenly distributed between the upper and lower halves of BL stage 2, all of the switches in the upper half can be set without regard to the value of the 3 MSB. If the same 3 MSBs exist on a switch in stage BL stage 2, that switch can be set without regard to the 3 MSBs since the upper output line always goes to an even switch and the lower output line always goes to an odd switch. Additionally, since only 2 outputs from BL stage 2 have the same 3 MSBs, no other switch will have a constraint based on these 3 MSBs.

These cases help eliminate 3 MSB conflicts but they don't guarantee a CCP in CC stage 0. So when one tries to provide for these scenarios, more generalized cases are missed. The goal is to find the generalized cases that always succeed. As stated earlier, when specific conditions are added to the heuristic, one has to make sure that probability of success increases and does not decrease.

Non-locked Pair Constraints

When there were locked pairs, the heuristic checked for 2 MSB and 3 MSB conflicts. There are ways that non-locked pairs combine that will cause conflicts. The heuristic could be modified to add checks for cases when non-locked pairs cause conflicts. Unfortunately, this involves checking numerous combinations of possible switch settings.

Backtracking

The switches are reset within a stage, and a concerted effort has been made to avoid backtracking from a current stage to a previous stage. Future work could involve a backtracking scheme as long as the time complexity of that scheme is considered and minimized.

Iterations

Efforts were made to run multiple iterations of the same input permutation as long as that permutation failed to produce a CCP in CC stage 0. These efforts were abandoned because the BL stage 0 switch settings were reset by the user for every iteration after the first iteration failed. The user's switch settings were random and not based on known conditions. The idea was simply "the other setting didn't work" so "try a different setting". The goal is to identify and control the conditions that allow a permutation to succeed, not stumble on a successful pass through the network. If iterations are to be explored, one must define a consistent repeatable process. One must define how many iterations to run and what stages to repeat. Running multiple iterations is similar to developing a backtracking scheme and the time complexity must also be analyzed and taken into consideration.

Permutations

One could exhaustively run all 16! permutations through this heuristic discover the exact number of success and failures. A more useful exercise would be to analyze the arrangement of the permutations that succeed and those that fail. This exercise could lead to additional necessary and sufficient conditions.

One way to generate permutations is in lexicographical order. When permutations were generated in this manner, there were 100 successes and 0 failures for the first 100 generated permutations. A comparison can be made to the Table 1 results for 100 uniformly random permutations, which had a success rate of just over 50%. This demonstrates that the more organized the input permutation, the more likely the chance of success. The more organized input permutations have fewer constraints as they pass through the BL stages. As more permutations were generated using the lexicographical algorithm, the percentage of successes decreased. There was a 90% success rate for the first 1,000,000 permutations. There was a 75% success rate for the 2 millionth through the 3 millionth generated permutations. As the arrangement of the permutations becomes more random, the probability of success decreases. The open question is what constraints are generated by randomly arranged permutations and how can conflicts due to these constraints be avoided. This is an important question to answer because the randomly arranged permutations make up the majority of the N! permutations.

Scalable Program

One final area for future work is to develop a scalable program. If the program for 16 inputs is scalable, one sets the foundation for not only proving the rearrangeability of a 16-input SE network but for proving the rearrangeability of the SE network for all $N=2^n$ inputs.

APPENDIX A

CONTROL HEURISTIC TO SET SWITCHES IN BASELINE STAGES

```
Initialize N = 16

Generate random input permutation

Initialize current input line
Initialize current output line

Process input permutation
     Validate inputs include 0 - 15 inclusively

Pass inputs through BL stages 0 - 2
     Determine BL stage 0 switch order
     Determine BL stage 0 switch settings
          Enforce condition 1
     Get BL stage 0 outputs

     Determine BL stage 1 switch order
     Set BL stage 0 outputs as BL stage 1 inputs
     Set BL stage 1 switches
       Enforce condition 2
       Enforce 2 MSB locked pair constraint
       Enforce 3 MSB locked pair constraint
       If conflict
          While position 1 switch not reset
               Reset current switch
                    If conflict
                         Unset switch
                         Reset previous switch
                    Else set next switch
       If no conflicts, Get BL stage 1 outputs
       Else exit

     Determine BL stage 2 switch order
     Set BL stage 1 outputs as BL stage 1 inputs
     Set BL stage 2 switches
       Enforce CCP MSB requirement
       Enforce CCP 2 MSB requirement
       Enforce CCP 3 MSB requirement
       If conflict
          While position 1 switch not reset
               Reset current switch
                    If conflict
                         Unset switch
                         Reset previous switch
                    Else set next switch
```

```
    If no conflicts, Get BL stage 2 outputs
    Else exit

  Determine CC stage 0 switch order
  Set BL stage 2 outputs as CC stage 0 inputs

Print results
```

APPENDIX B

CONTROL HEURISTIC RESULTS

*********RESULTS - CATEGORY A, #1**************

Here is your input permutation:
11 14 1 2 5 6 0 4 9 13 7 15 10 12 3 8

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

CCP Req1 PASSED for CC Stage 0: Even distribution of MSB on every switch
CCP Req2 PASSED for CC Stage 0: All perm. of 2MSB on (0,4), (1,5), (2,6), and (3,7)
CCP Req3 PASSED for CC Stage 0: All perm. of 3MSB on (0,4,2,6) and (1,5,3,7)

CONGRATULATIONS! You have a CCP!
****

BL Stage 0	BL Stage 1	BL Stage 2	CC Stage 0
1011 0	1011 0	1011 0	1011 0
1110 =	0001 =	0000 =	0111 *
0001 4	0101 4	0111 4	0000 1
0010 =	0000 X	1010 =	1010 *
0101 2	1001 2	0001 1	0001 2
0110 =	0111 X	0101 =	1001 *
0000 6	1010 6	1001 5	0101 3
0100 =	1000 =	1000 =	1000 *
1001 1	1110 1	1110 2	1110 4
1101 =	0010 =	0110 =	0011 *
0111 5	0110 5	1101 6	0110 5
1111 =	0100 =	0011 X	1101 *
1010 3	1101 3	0010 3	0100 6
1100 =	1111 =	0100 X	1100 *
0011 7	1100 7	1111 7	0010 7
1000 X	0011 X	1100 X	1111 *

*********RESULTS - CATEGORY A, #2**************

Here is your input permutation:
15 4 0 10 11 12 6 1 7 8 2 5 3 13 9 14

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

CCP Req1 PASSED for CC Stage 0: Even distribution of MSB on every switch
CCP Req2 PASSED for CC Stage 0: All perm. of 2MSB on (0,4), (1,5), (2,6), and (3,7)
CCP Req3 PASSED for CC Stage 0: All perm. of 3MSB on (0,4,2,6) and (1,5,3,7)

CONGRATULATIONS! You have a CCP!
****

```
  BL         BL         BL         CC
 Stage 0    Stage 1    Stage 2    Stage 0

 1111 0     1111 0     1111 0     1111 0
 0100 =     0000 =     0110 =     0111 *

 0000 4     1011 4     0111 4     0110 1
 1010 =     0110 X     1101 =     1101 *

 1011 2     0111 2     0000 1     0000 2
 1100 =     0010 =     1011 =     1001 *

 0110 6     1101 6     0010 5     1011 3
 0001 =     1001 =     1001 X     0010 *

 0111 1     0100 1     1010 2     1010 4
 1000 =     1010 X     0001 =     0011 *

 0010 5     1100 5     1000 6     0001 5
 0101 =     0001 X     0011 X     1000 *

 0011 3     1000 3     0100 3     1100 6
 1101 X     0101 =     1100 X     0101 *

 1001 7     0011 7     0101 7     0100 7
 1110 =     1110 =     1110 =     1110 *
```

```
*********RESULTS - CATEGORY A, #3**************

Here is your input permutation:
0 10 3 8 15 6 4 9 14 5 7 12 13 11 2 1

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and
lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and
lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

CCP Req1 PASSED for CC Stage 0:  Even distribution of MSB on every
switch
CCP Req2 PASSED for CC Stage 0:  All perm. of 2MSB on (0,4), (1,5),
(2,6), and (3,7)
CCP Req3 PASSED for CC Stage 0:  All perm. of 3MSB on (0,4,2,6) and
(1,5,3,7)

CONGRATULATIONS! You have a CCP!
*********************************

   BL        BL        BL        CC
 Stage 0   Stage 1   Stage 2   Stage 0

 0000 0    0000 0    0000 0    0000 0
 1010 =    1000 =    1111 =    1110 *

 0011 4    1111 4    1110 4    1111 1
 1000 X    0100 =    0010 =    0010 *

 1111 2    1110 2    1000 1    1000 2
 0110 =    0111 =    0100 =    0111 *

 0100 6    1101 6    0111 5    0100 3
 1001 =    0010 X    1101 =    1101 *

 1110 1    1010 1    1010 2    1010 4
 0101 =    0011 =    0110 =    0101 *

 0111 5    0110 5    0101 6    0110 5
 1100 =    1001 =    1011 =    1011 *

 1101 3    0101 3    0011 3    0011 6
 1011 =    1100 =    1001 =    1100 *

 0010 7    1011 7    1100 7    1001 7
 0001 =    0001 =    0001 =    0001 *
```

```
*********RESULTS - CATEGORY A, #4**************

Here is your input permutation:
6 13 12 3 1 10 15 8 2 5 14 4 11 7 0 9

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and
lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and
lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

CCP Req1 PASSED for CC Stage 0:  Even distribution of MSB on every
switch
CCP Req2 PASSED for CC Stage 0:  All perm. of 2MSB on (0,4), (1,5),
(2,6), and (3,7)
CCP Req3 PASSED for CC Stage 0:  All perm. of 3MSB on (0,4,2,6) and
(1,5,3,7)

CONGRATULATIONS! You have a CCP!
*********************************

   BL         BL         BL         CC
 Stage 0    Stage 1    Stage 2    Stage 0

 0110 0     0110 0     0110 0     0110 0
 1101 =     1100 =     1111 =     1110 *

 1100 4     0001 4     1110 4     1111 1
 0011 =     1111 X     0000 =     0000 *

 0001 2     0010 2     1100 1     0001 2
 1010 =     1110 X     0001 X     1011 *

 1111 6     1011 6     0010 5     1100 3
 1000 =     0000 X     1011 X     0010 *

 0010 1     1101 1     0011 2     0011 4
 0101 =     0011 X     1010 =     1001 *

 1110 5     1010 5     0101 6     1010 5
 0100 =     1000 =     1001 X     0101 *

 1011 3     0101 3     1101 3     1101 6
 0111 =     0100 =     1000 =     0100 *

 0000 7     0111 7     0100 7     1000 7
 1001 =     1001 X     0111 =     0111 *
```

*********RESULTS - CATEGORY A, #5**************

Here is your input permutation:
7 3 11 14 12 13 4 9 10 5 8 2 15 6 1 0

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

CCP Req1 PASSED for CC Stage 0: Even distribution of MSB on every switch
CCP Req2 PASSED for CC Stage 0: All perm. of 2MSB on (0,4), (1,5), (2,6), and (3,7)
CCP Req3 PASSED for CC Stage 0: All perm. of 3MSB on (0,4,2,6) and (1,5,3,7)

CONGRATULATIONS! You have a CCP!
****

BL Stage 0	BL Stage 1	BL Stage 2	CC Stage 0
0111 0	0111 0	0111 0	0111 0
0011 =	1011 =	1100 =	1010 *
1011 4	1100 4	1010 4	1100 1
1110 =	0100 =	0110 =	0110 *
1100 2	1010 2	1011 1	0100 2
1101 =	1000 =	0100 X	1000 *
0100 6	0110 6	1000 5	1011 3
1001 =	0001 =	0001 =	0001 *
1010 1	0011 1	0011 2	0011 4
0101 =	1110 =	1001 =	1111 *
1000 5	1101 5	0010 6	1001 5
0010 =	1001 X	1111 X	0010 *
1111 3	0101 3	1110 3	1101 6
0110 X	0010 X	1101 X	0000 *
0001 7	1111 7	0101 7	1110 7
0000 =	0000 =	0000 X	0101 *

```
*********RESULTS - CATEGORY B, #1**************

Here is your input permutation:
2 9 11 13 5 6 7 3 12 8 15 4 1 0 10 14

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and
lower quadrants

Cond1 FAILED for BL Stage 2: Even distribution of MSB in upper and
lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
*********************************

  BL         BL         BL         CC
 Stage 0    Stage 1    Stage 2    Stage 0

 0010 0     0010 0     1011 0      *** 0
 1001 =     1011 X     0111 X      *** *

 1011 4     0101 4     1111 4      *** 1
 1101 =     0111 X     0001 =      *** *

 0101 2     1100 2     0010 1      *** 2
 0110 =     1111 X     0101 X      *** *

 0111 6     0001 6     1100 5      *** 3
 0011 =     1010 =     1010 X      *** *

 1100 1     1001 1     1101 2      *** 4
 1000 =     1101 X     0011 X      *** *

 1111 5     0110 5     0100 6      *** 5
 0100 =     0011 X     1010 X      *** *

 0001 3     1000 3     1001 3      *** 6
 0000 =     0100 X     0110 X      *** *

 1010 7     0000 7     1000 7      *** 7
 1110 =     1110 =     1110 =      *** *
```

*********RESULTS - CATEGORY B, #2**************

Here is your input permutation:
6 2 1 5 10 3 12 15 13 7 14 4 0 9 8 11

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
****

```
  BL          BL          BL          CC
 Stage 0     Stage 1     Stage 2     Stage 0

 0110 0      0110 0      0110 0       *** 0
 0010 =      0001 =      1010 X       *** *

 0001 4      1010 4      1110 4       *** 1
 0101 =      1100 =      0000 X       *** *

 1010 2      0111 2      0001 1       *** 2
 0011 =      1110 X      1100 X       *** *

 1100 6      0000 6      0111 5       *** 3
 1111 =      1000 =      1000 =       *** *

 1101 1      0010 1      0010 2       *** 4
 0111 X      0101 =      1111 X       *** *

 1110 5      0011 5      0100 6       *** 5
 0100 =      1111 X      1001 X       *** *

 0000 3      1101 3      0101 3       *** 6
 1001 =      0100 X      0011 X       *** *

 1000 7      1001 7      1101 7       *** 7
 1011 =      1011 =      1011 =       *** *
```

*********RESULTS - CATEGORY B, #3**************

Here is your input permutation:
10 15 7 13 4 0 3 11 6 1 5 12 14 8 2 9

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
****

```
  BL         BL         BL         CC
 Stage 0    Stage 1    Stage 2    Stage 0

 1010 0     1010 0     1010 0      *** 0
 1111 =     0111 =     0100 X      *** *

 0111 4     0100 4     0110 4      *** 1
 1101 =     1011 =     1110 X      *** *

 0100 2     0110 2     0111 1      *** 2
 0000 =     0101 =     1011 =      *** *

 0011 6     1110 6     0101 5      *** 3
 1011 X     1001 =     1001 =      *** *

 0110 1     1111 1     1111 2      *** 4
 0001 =     1101 =     0000 =      *** *

 0101 5     0000 5     0001 6      *** 5
 1100 =     0011 =     1000 =      *** *

 1110 3     0001 3     1101 3      *** 6
 1000 =     1100 =     0011 =      *** *

 0010 7     1000 7     1100 7      *** 7
 1001 X     0010 =     0010 =      *** *
```

```
*********RESULTS - CATEGORY B, #4**************

Here is your input permutation:
12 6 15 3 1 14 9 4 0 7 10 2 11 8 5 13

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and
lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and
lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
*********************************

  BL        BL        BL        CC
 Stage 0   Stage 1   Stage 2   Stage 0

 1100 0    1100 0    1100 0     *** 0
 0110 =    0011 =    0001 X     *** *

 1111 4    0001 4    0000 4     *** 1
 0011 X    1001 =    1011 X     *** *

 0001 2    0000 2    0011 1     *** 2
 1110 =    1010 =    1001 X     *** *

 1001 6    1011 6    1010 5     *** 3
 0100 =    0101 =    0101 X     *** *

 0000 1    0110 1    0110 2     *** 4
 0111 =    1111 =    1110 X     *** *

 1010 5    1110 5    0111 6     *** 5
 0010 =    0100 =    1000 =     *** *

 1011 3    0111 3    1111 3     *** 6
 1000 =    0010 =    0100 =     *** *

 0101 7    1000 7    0010 7     *** 7
 1101 =    1101 =    1101 =     *** *
```

*********RESULTS - CATEGORY B, #5**************

Here is your input permutation:
8 1 7 0 2 5 10 14 6 15 13 3 4 9 12 11

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Cond1 PASSED for BL Stage 2: Even distribution of MSB in upper and lower sub-quadrants
Cond2 PASSED for BL Stage 2: Pair of 2MSB on (0,4,2,6) and (1,5,3,7)
Locked pair conflicts AVOIDED in BL Stage 2

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
****

```
  BL          BL          BL          CC
 Stage 0     Stage 1     Stage 2     Stage 0

 1000 0      1000 0      1000 0       *** 0
 0001 =      0111 =      0010 X       *** *

 0111 4      0010 4      0110 4       *** 1
 0000 =      1010 =      1100 X       *** *

 0010 2      0110 2      0111 1       *** 2
 0101 =      1101 =      1010 =       *** *

 1010 6      0100 6      1101 5       *** 3
 1110 =      1100 X      0100 X       *** *

 0110 1      0001 1      0001 2       *** 4
 1111 =      0000 =      0101 =       *** *

 1101 5      0101 5      1111 6       *** 5
 0011 =      1110 =      1001 =       *** *

 0100 3      1111 3      0000 3       *** 6
 1001 =      0011 =      1110 =       *** *

 1100 7      1001 7      0011 7       *** 7
 1011 =      1011 =      1011 =       *** *
```

*********RESULTS - CATEGORY C, #1**************

Here is your input permutation:
8 1 3 13 10 4 0 2 6 11 7 14 15 9 5 12

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Sorry! BL Stage 1 switches could not be set
to generate acceptable inputs for BL Stage 2.

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
****

BL Stage 0	BL Stage 1	BL Stage 2	CC Stage 0
1000 0	1000 0	*** 0	*** 0
0001 =	0011 X	*** *	*** *
0011 4	1010 4	*** 4	*** 1
1101 =	0000 X	*** *	*** *
1010 2	0110 2	*** 1	*** 2
0100 =	0111 X	*** *	*** *
0000 6	1111 6	*** 5	*** 3
0010 =	1100 X	*** *	*** *
0110 1	0001 1	*** 2	*** 4
1011 =	1101 X	*** *	*** *
0111 5	0100 5	*** 6	*** 5
1110 =	0010 X	*** *	*** *
1111 3	1011 3	*** 3	*** 6
1001 =	1110 =	*** *	*** *
0101 7	1001 7	*** 7	*** 7
1100 X	0101 X	*** *	*** *

*********RESULTS - CATEGORY C, #2**************

Here is your input permutation:
14 9 0 13 12 4 1 5 11 3 10 7 6 8 2 15

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Sorry! BL Stage 1 switches could not be set
to generate acceptable inputs for BL Stage 2.

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
****

BL Stage 0	BL Stage 1	BL Stage 2	CC Stage 0
1110 0	1110 0	*** 0	*** 0
1001 =	0000 X	*** *	*** *
0000 4	1100 4	*** 4	*** 1
1101 =	0001 X	*** *	*** *
1100 2	1011 2	*** 1	*** 2
0100 =	0111 =	*** *	*** *
0001 6	0110 6	*** 5	*** 3
0101 =	1111 X	*** *	*** *
1011 1	1001 1	*** 2	*** 4
0011 =	1101 X	*** *	*** *
1010 5	0100 5	*** 6	*** 5
0111 X	0101 X	*** *	*** *
0110 3	0011 3	*** 3	*** 6
1000 =	1010 X	*** *	*** *
0010 7	1000 7	*** 7	*** 7
1111 X	0010 X	*** *	*** *

```
*********RESULTS - CATEGORY C, #3**************

Here is your input permutation:
2 11 9 8 3 7 0 6 4 1 12 14 13 5 15 10

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and
lower quadrants

Sorry! BL Stage 1 switches could not be set
to generate acceptable inputs for BL Stage 2.

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
*********************************

   BL        BL        BL        CC
 Stage 0   Stage 1   Stage 2   Stage 0

 0010 0    0010 0     *** 0     *** 0
 1011 =    1001 X     *** *     *** *

 1001 4    0011 4     *** 4     *** 1
 1000 =    0000 X     *** *     *** *

 0011 2    0100 2     *** 1     *** 2
 0111 =    1100 X     *** *     *** *

 0000 6    1101 6     *** 5     *** 3
 0110 =    1111 X     *** *     *** *

 0100 1    1011 1     *** 2     *** 4
 0001 =    1000 X     *** *     *** *

 1100 5    0111 5     *** 6     *** 5
 1110 =    0110 X     *** *     *** *

 1101 3    0001 3     *** 3     *** 6
 0101 =    1110 X     *** *     *** *

 1111 7    0101 7     *** 7     *** 7
 1010 =    1010 =     *** *     *** *
```

*********RESULTS - CATEGORY C, #4**************

Here is your input permutation:
1 13 14 3 6 12 7 2 15 4 0 10 5 9 11 8

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and lower quadrants

Sorry! BL Stage 1 switches could not be set
to generate acceptable inputs for BL Stage 2.

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
****

BL Stage 0	BL Stage 1	BL Stage 2	CC Stage 0
0001 0	0001 0	*** 0	*** 0
1101 =	1110 X	*** *	*** *
1110 4	0110 4	*** 4	*** 1
0011 =	0111 X	*** *	*** *
0110 2	1111 2	*** 1	*** 2
1100 =	0000 X	*** *	*** *
0111 6	1001 6	*** 5	*** 3
0010 =	1011 X	*** *	*** *
1111 1	1101 1	*** 2	*** 4
0100 =	0011 X	*** *	*** *
0000 5	1100 5	*** 6	*** 5
1010 =	0010 =	*** *	*** *
0101 3	0100 3	*** 3	*** 6
1001 X	1010 X	*** *	*** *
1011 7	0101 7	*** 7	*** 7
1000 =	1000 =	*** *	*** *

```
*********RESULTS - CATEGORY C, #5**************

Here is your input permutation:
2 7 0 12 5 15 13 11 8 1 9 4 14 3 6 10

BL Stage 0 inputs are valid: All numbers from 0 to 15 inclusively

Cond1 PASSED for BL Stage 1: Even distribution of MSB in upper and
lower quadrants

Sorry! BL Stage 1 switches could not be set
to generate acceptable inputs for BL Stage 2.

Sorry! BL Stage 2 switches could not be set
to generate acceptable inputs for CC Stage 0.

Sorry! You do not have a CCP.
*********************************

   BL         BL         BL         CC
 Stage 0    Stage 1    Stage 2    Stage 0

 0010 0     0010 0      *** 0      *** 0
 0111 =     0000 X      *** *      *** *

 0000 4     1111 4      *** 4      *** 1
 1100 =     1101 X      *** *      *** *

 0101 2     1000 2      *** 1      *** 2
 1111 X     0100 X      *** *      *** *

 1101 6     1110 6      *** 5      *** 3
 1011 =     0110 =      *** *      *** *

 1000 1     0111 1      *** 2      *** 4
 0001 =     1100 =      *** *      *** *

 1001 5     0101 5      *** 6      *** 5
 0100 X     1011 X      *** *      *** *

 1110 3     0001 3      *** 3      *** 6
 0011 =     1001 X      *** *      *** *

 0110 7     0011 7      *** 7      *** 7
 1010 =     1010 =      *** *      *** *
```

REFERENCES

[1] H.S.Stone, "Parallel Processing with the Perfect Shuffle," *IEEE Transaction on Computers*, Vol. C-20, February 1971, pp. 153-161.

[2] A.Varma and C.S.Raghavendra, "Rearrangeability of Multistage Shuffle/Exchange Networks," in *Proceedings 14th Annual Symposium Computer Architecture*, June 1987, pp. 154-162.

[3] D.H.Lawrie, "Access and Alignment of Data in an Array Processor," *IEEE Transactions on Computers*, Vol. C-24, Dec. 1975. pp. 1145-1155.

[4] C.P.Kruskal and M.Snir, "A Unified Theory of Interconnection Network Structure," *Theoretical Computer Science*, Vol. 48, 1986, pp. 75-94.

[5] T.Lang, "Interconnections between Processors and Memory Modules using the Shuffle Exchange Network," *IEEE Transactions on Computers*, Vol. C-25, No. 5, May 1976 PP. 175-189.

[6] V.E.Benes, "Proving the Rearrangeability of Connecting Networks by Group Calculations," *Bell System Technical Journal*, 45, 1975, pp. 421-434.

[7] H.J.Siegel, "Partitionable SIMD Computer System Interconnection Network Universality," *Proceedings 16th Annual Allerton Conference on Communications, Control, and Computing*, October 1978, pp. 586-595.

[8] D.S.Parker, "Notes on Shuffle/Exchange-Type Switching Networks," *IEEE Transactions on Computers*, Vol. C-29, No. 3, March 1980, pp. 213-222.

[9] C.L.Wu, T.Y.Feng, "The Universality of the Shuffle-Exchange Network," *IEEE Transactions on Computers*, Vol. C-30, No. 5, May 1981, pp. 324-332.

[10] C.S.Raghavendra and A.Varma, "Rearrangeability of the Five Stage Shuffle-Exchange Network for N=8," *IEEE Transactions on Communications*, Vol.Com-35, Aug. 1987, pp. 808-812.

[11] C.L.Wu and T.Y.Feng, "On a Class of Multistage Interconnection Networks," *IEEE Transactions on Computing*, Vol. C-29, Aug 1980, pp. 696-702.

[12] A.Y.Oruc and M.Y.Oruc, "Equivalence Relations Among Interconnection Networks," *Journal of Parallel and Distributed Computing*, Vol. 2, 1985, pp. 30-49.

[13] D.E. Knuth, "The Art of Computer Programming", Vol. 2, Seminumerical Algorithms, Third Edition, Addison-Wesley, Reading, MA, 1997, p. 139.