Hardware Implementation Of Processor Allocator For Mesh Connected Chip Multiprocessors

Rana Sangram Reddy Marri
University of Nevada, Las Vegas, rana99sangram@gmail.com

Follow this and additional works at: https://digitalscholarship.unlv.edu/thesesdissertations
Part of the Electrical and Computer Engineering Commons, and the Hardware Systems Commons

Repository Citation
https://digitalscholarship.unlv.edu/thesesdissertations/1754

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Theses, Dissertations, Professional Papers, and Capstones by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
HARDWARE IMPLEMENTATION OF PROCESSOR_ALLOCATOR FOR MESH CONNECTED CHIP MULTIPROCESSORS

By

Rana Sangram Reddy Marri

Bachelor of Technology in Electronics and Communication Engineering
Jawaharlal Nehru Technological University, India
May 2009

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering
Howard R. Hughes College of Engineering
The Graduate College

University of Nevada, Las Vegas
December 2012
THE GRADUATE COLLEGE

We recommend the thesis prepared under our supervision by

Rana Sangram Reddy Marri

entitled

Hardware Implementation of Processor Allocator for Mesh Connected Chip Multiprocessors

be accepted in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering
Department of Electrical and Computer Engineering

Henry Selvaraj, Ph.D. Committee Chair

Emma Regentova Ph.D. Committee Member

Yingtao Jiang, Ph.D. Committee Member

Laxmi Gewali, Ph.D. Graduate College Representative

Tom Piechota, Ph.D., Interim Vice President for Research &
Dean of the Graduate College

December 2012
ABSTRACT

HARDWARE IMPLEMENTATION OF PROCESSOR ALLOCATOR FOR MESH CONNECTED CHIP MULTIPROCESSORS

by

Rana Sangram Reddy Marri

Dr. Henry Selvaraj, Examination Committee Chair
Professor of Electrical and Computer Engineering
University of Nevada, Las Vegas

The advancements in the semiconductor process technology and the current demand for highly parallel computing has led to the advent of Chip Multiprocessors (CMPs). CMP is the integration of two or more independent processor cores, which can read and execute program instructions, on to a single integrated circuit die. CMPs are the main computing platforms for research and development in parallel and high performance computing environments. They offer minimum inter-core communication latencies as the processor cores are present on a single chip.

The Operating System (OS) plays a key role in using a CMP effectively. The OS should support a multi-user environment in which the jobs are executed in parallel on different cores. This is handled by the processor management system of the OS. The Processor Management System consists of Job Scheduler (JS) and Processor Allocator (PA). The JS aligns the jobs in a queue in an order which is determined by the scheduling
policy employed and thus specifying the job that is to be executed next. The PA deals with the selection of appropriate set of processors to execute the job scheduled by the job scheduler. Efficient design of a PA is crucial if one is to harness the full computational power of a CMP in large parallel computing systems.

This thesis deals with the processor allocation part of the processor management system. The motive of this thesis is the hardware implementation of a PA for a mesh-connected CMP. The PA is implemented and a synthesis report is presented which shows the amount of logic utilized. Many contiguous and non-contiguous allocation strategies have been proposed for mesh networks in the recent years. The Improvised First Fit algorithm is used to select the appropriate set of processors for executing an incoming job in this hardware implementation. This algorithm is a contiguous allocation algorithm and has complete sub-mesh recognition ability and uses a bit-map approach. The JS is assumed to be employing a First Come First Serve (FCFS) policy to schedule the jobs. This thesis also acts as the basis for the hardware implementation of PA that uses other allocation algorithms in different topologies.
ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor and thesis advisory committee chair, Dr. Henry Selvaraj. His continuous support throughout my master’s program here at the University of Nevada, Las Vegas has been tremendous. He is not only an excellent professor but also a wonderful human being. He has been a mentor to me and was always there to help me out whenever I was in a fix. It would have been difficult for me to survive in this foreign land if it was not for Dr. Selvaraj’s support and motivation. He always gave me the freedom to come up with any new ideas. I lack the words to express how grateful I am to Dr. Selvaraj.

I would like to thank the members of my thesis advisory committee Dr. Emma Regentova, Dr. Yingtao Jiang and Dr. Laxmi Gewali for finding time from their busy schedules, accepting to be a part of my thesis advisory committee and for their valuable remarks.

I am grateful to god for giving me the most wonderful parents and brother who supported me at all times without expecting anything but the best for me. I would like to express special thanks to Dr. Satish Bhatnagar who has been very encouraging. I can never forget the affection Dr. Bhatnagar has shown me during my initial period of stay in the USA. I thank all my friends here and back home who helped me at every stage of my life to become a better person.
Table of Contents

ABSTRACT.................................................................................................................................iii

ACKNOWLEDGMENTS ..............................................................................................................v

TABLE OF CONTENTS............................................................................................................vi

LIST OF FIGURES...................................................................................................................viii

CHAPTER 1 INTRODUCTION....................................................................................................1

1.1 Thesis Overview..................................................................................................................4

CHAPTER 2 CHIP MULTIPROCESSORS...................................................................................5

2.1 Emergence of Chip Multiprocessors ..................................................................................5

2.2 Advantages of CMP .........................................................................................................11

2.3 Challenges in Chip Multiprocessors ................................................................................12

2.4 Types of CMP ..................................................................................................................14

CHAPTER 3 PROCESSOR MANAGEMENT SYSTEM.............................................................15

3.1 Processor Management in Multi-core Processors .............................................................15

3.2 Processor Allocation Algorithms ....................................................................................18

3.3 Job scheduling and Processor Allocation .........................................................................20

3.4 Network on Chip (NoC) ..................................................................................................21
CHAPTER 4 PROCESSOR ALLOCATION

4.1 Processor Allocation Strategy Used ..........................................................25

4.1.1 K-ary 2-mesh .........................................................................................25

4.1.2 Bitmap Approach ..................................................................................31

4.1.3 First Fit Algorithm and Improved First Fit Algorithm ..........................34

4.2 Processor Allocator Architecture ................................................................37

4.2.1 Processor Allocator Architecture Used ..................................................39

CHAPTER 5 HARDWARE IMPLEMENTATION ..................................................41

5.1 Implementation on ML403 Evaluation Platform .......................................44

5.2 Implementation on Spartan 3E starter Kit ..................................................53

CHAPTER 6 CONCLUSIONS AND FUTURE DIRECTIONS ...............................63

6.1 Future Directions .......................................................................................65

BIBLIOGRAPHY ...............................................................................................67

VITA .....................................................................................................................73
List of Figures

Figure 2.1 Von Neumann architecture ................................................................. 6

Figure 2.2 Transistor count over time ................................................................. 8

Figure 2.3 Clock frequency over time ................................................................. 9

Figure 2.4 CMP architecture ............................................................................. 11

Figure 2.5 Speed up vs. Number of processors .................................................. 12

Figure 3.1 Illustration of 4x4 grid structured NoC ............................................. 16

Figure 3.2 Processor Management System ....................................................... 17

Figure 3.3 A 7X7 2D Mesh NoC ....................................................................... 22

Figure 3.4 A 4X4 mesh NoC with 16 resources .................................................. 23

Figure 4.1 Mesh M(w,h) with width “w” and height “h” ..................................... 26

Figure 4.2 Mesh M(8,8) with row numbers and column numbers ....................... 27

Figure 4.3 Submesh S(<2,3><4,4>) ................................................................... 28

Figure 4.4 Bitmap showing the status (busy/free) of the processors .................... 29

Figure 4.5 Reject area with respect to the job J(3,2) ........................................... 30

Figure 4.6 Free node, Busy node, Sink, Coverage area with respect to the job J(2,3).... 31
Figure 4.7 Bitmap of the corresponding Mesh representing free and busy processors…32

Figure 4.8 Coverage area.................................................................33

Figure 4.9 The PA architecture based on the bitmap approach..................38

Figure 4.10 The Processor Allocator Model used in this thesis..................40

Figure 5.1 The Processor Allocator module used for Hardware implementation……43

Figure 5.2 Virtex 4 ML403 Evaluation Platform.........................................45

Figure 5.3 Device Utilization Summary using a ML403 Evaluation Platform……47

Figure 5.4 Translation report while implementing on ML403 Evaluation Platform……48

Figure 5.5 Screenshot showing Detailed reports in ML403 Evaluation platform ……49

Figure 5.6 Package while implementing on ML403 Evaluation Platform…………50

Figure 5.7 Place and Route completion in ML403 Evaluation Platform…………51

Figure 5.8 Screenshot showing the Bitstream generation in ML403 board…………52

Figure 5.9 Spartan 3E FPGA starter kit.................................................54

Figure 5.10 Device Utilization Summary using a Xilinx Spartan 3E Starter Kit……56

Figure 5.11 Package schematic while implementing on Xilinx Spartan 3E Starter Kit…57

Figure 5.12 Design schematic while implementing on Xilinx Spartan 3E Starter Kit….58

Figure 5.13 Clock Net while implementing on Xilinx Spartan 3E Starter Kit………59
Figure 5.14 Completion of Place and Route for Xilinx Spartan 3E Starter Kit…………..60

Figure 5.15 Completion of Bitstream for Xilinx Spartan 3E Starter Kit……………………61
CHAPTER 1
INTRODUCTION

The Microprocessors have been there for about four decades now. Because of the advancement of technology, each day, there have been numerous improvisations to the architectures and their functioning. Increasing the clock frequency and increasing the logic density has resulted in the increased performance of a microprocessor. Hence extensive research has always been done in these areas to increase the efficiency of a microprocessor [2].

In 1965, G E Moore predicted, “The number of transistors on a given piece of silicon of an integrated circuit would increase exponentially, doubling every couple of years” [3]. This is called “Moore’s law”. This law can be observed when the microprocessors from the year 1971 (the year of their introduction) are compared to that of 2011. The number of transistors has increased exponentially from couple of thousands to thousands of millions [5]. This has resulted in the reduction of cost and increased reliability. Because of the increase in the chip density, manufacturers started to integrate more and more components on a single die which has given rise to the concept of System on Chip (SoC).

Complex systems such as processing units, various peripherals, dedicated bus structures and storage elements can be integrated on a single die with the help of System on Chip technology [4]. SoC design methodology results in chips with less power consumption, more reliability and less cost when compared to the individual chip systems that they replace. The static and dynamic power dissipations play a key role in the chip
design. The dynamic power dissipation depends on the frequency and the static power dissipation depends on the number of transistors on the chip. This implies that the Moore’s law limits the processing speeds that can be reached. Moreover, the heat dissipated and the electromagnetic interference will put a cap on the transistor density on a chip. This has led to the introduction of Chip Multiprocessors (CMPs) [2].

In parallel computing, multiple processors work together with a goal to solve a given problem. The communication between these processors is based either on shared memory or distributed memory models. Multiprocessors have a shared memory architecture where the processors communicate via shared memory. CMP is the integration of multiple independent processor cores on a single chip. If these cores are identical then the resultant CMP is referred to as a Homogenous CMP. If the cores on a CMP are different then the resultant CMP is referred to as a Heterogeneous CMP. CMP plays a prominent role in faster, parallel and high performance computing while mitigating the heat dissipation due to increase in the clock frequencies. There is a lot of research going on in designing effective CMP to extract maximum throughput by using the on-chip resources effectively.

The Operating System (OS) is crucial in managing the processes efficiently. The OS should allow multi-user environment to execute the processes in parallel and increase the throughput. The Processor Management System of the OS handles job scheduling and processor allocation. Job scheduler (JS) and Processor allocator (PA) are two components of the processor management system. The JS arranges the jobs in an order, thus specifying which job has to be serviced next. The PA allocates the processing resources to the incoming job that has to be serviced. In the CMPs it is critical to design a PA
effectively so that it uses the processing elements and other resources efficiently in order to get optimal throughput. The on-chip real estate in the CMPs should be used very efficiently especially in the case where the PA is to be integrated on a CMP. The internal architecture of the PA depends on the allocation algorithm used. Several contiguous and non contiguous allocation schemes have been proposed to acquire optimal performance in the CMPs. An effective allocation algorithm should make the maximum use of the resources effectively, offer less allocation and de-allocation overheads and should have sub-mesh recognition ability. In this thesis, an Improved First Fit (IFF) algorithm which uses a bitmap approach has been used for allocation by a PA.

The topology of the processing elements in a CMP also plays a role in coming up with an effective allocation strategy. Different allocation strategies exploiting different network topologies have been proposed. Most of the current high performance parallel computing applications are based on mesh connected multiprocessors systems. This thesis assumes a mesh connected CMP which implies that the processing elements in the CMP are connected in a mesh topology.

This thesis gives a brief introduction to the CMPs and the importance of PA and the allocation algorithms in extracting high performance from a CMP. The main objective of this thesis is the hardware implementation of the PA which uses an Improved First Fit allocation strategy based on the bitmap approach. The processing elements are assumed to be connected in a mesh fashion. The Virtex 4 ML403 Evaluation Platform and Spartan 3E starter Kit have been used for the implementation. The boards are equipped with Virtex-4 (XC4VFX12-FF668-10) and Spartan-3E (XC3S500E-4FG320C) FPGAs.
respectively. The PA is synthesized, implemented on the board and the synthesis results are presented.

### 1.1 THESIS OVERVIEW

This thesis gives a brief introduction to the Chip Multiprocessors. The thesis concentrates on the processor allocation part of the Processor Management System. The main objective of this thesis is the hardware implementation of the processor allocator in the mesh-connected Chip Multiprocessors.

Chapter 2 introduces the reader to Chip Multiprocessors, their advantages and challenges faced by them.

Chapter 3 discusses about the Processor Management System in Multi-core processors, the processor allocation strategies and a brief note on job scheduling and NoC.

Chapter 4 discusses about the topology of the connected processors taken into consideration, allocation strategy used and the processor allocator architecture.

Chapter 5 deals with the objective of this thesis which is the hardware implementation of the processor allocator in mesh-connected CMPs. The Synthesis reports showing the amount of logic utilized in both the boards used is also available in this chapter.

Chapter 6 gives the conclusions for this thesis and the future work that can be done.
2.1 Emergence of Chip Multiprocessors

Microprocessors have a long history in computer architecture. The inception of low cost processing units on integrated circuits (IC) brought a revelation in the modern computing world. The “Intel 4004” is the first microprocessor designed by Intel Corporation in 1971. It has a 4-bit parallel central processing unit (CPU) with 46 instructions on a single chip. It has 2300 transistors in 10 micron technology with a clock speed of 108 kHz with an instruction cycle time of 10.8 microseconds. The Intel 4004 was used in a Busicom calculator for arithmetic manipulations [6]. There have been tremendous improvisations in the architectures of the microprocessors in the quest for faster and smarter computing ever since their origin.

Modern microprocessors, because of features like pipelining and parallelism, can execute complex programs efficiently. The number of transistors that can be embedded on an IC in the modern microprocessors has increased from couple of thousands to thousands of millions compared to that of Intel 4004. But these microprocessors still used the Von Neumann architecture. Figure 2.1 shows the block diagram representing Von Neumann architecture.
As can be seen from the figure 2.1, Von Neumann architecture consists of three building blocks connected by the system bus. They are as follows:

- **Central processing Unit**:

  The primary components of the CPU are arithmetic and logical unit (ALU), control unit, registers and a program control (PC). Registers are fast working storage components in the CPU. ALU is responsible for arithmetic and logical operations. Control Unit decodes the instructions, controls the data flow and manages the execution of those
instructions and storing the results. PC is responsible for pointing to the address of the next instruction that is to be executed.

- Memory:

  This is the block responsible for the storage of program instructions, data, intermediate results and the final results.

- Input/output controllers:

  Input controller provides the means to enter data and instructions to the system. Output controller is the means to get the results from the system.

The instruction execution cycle in the Von Neumann architecture involves these steps:

1. Calculate the address of the instruction to be executed and prompt the main memory to read the instruction at that address

2. Fetch the instruction to the CPU

3. Decode and execute the instruction inside the CPU

4. Go to 1

The basic principle of this architecture is still followed but with each step highly optimized because of the advances in technology. There are a few factors affecting the performance of a microprocessor but the technological advances in these areas, viz. clock speed and the logic density on the chip have been prominent. The increase in clock frequency enables faster execution of instructions and more logic density implies more
functionality embedded in the chip. The Chip density has increased several folds since the first microprocessor. The number of transistors over time can be observed from the figure 2.2.

![CHIP COMPLEXITY](image)

**Figure 2.2 Transistor count over time [8]**

The number of transistors that can be fitted on a single silicon die over time has been in accordance with Moore’s law. In 1965, Gordon E. Moore, the co-founder of the Intel Corporation, predicted that number of transistors that can be incorporated on a given silicon die will double every two years. This is now called as the Moore’s law. However there are few limits to this prediction. There is a physical bound on the number of transistors that can be fitted in a given silicon die as the size of the transistor cannot be
reduced indefinitely. Also, there is power dissipation factor and limit on the clock speeds when dealing with such high density chips. The increase in clock speed results in the increase of the power consumed. The clock frequencies of the microprocessors over the past two decades can be seen in the figure 2.3.

![CLOCK FREQUENCY](image)

Figure 2.3 Clock frequency over time [8]

With the advancement in technology and tremendous increase in the number of transistors that can be accommodated on a given silicon die, the design methodology has changed from gate level perspective to integrating complex components onto a single chip. This paved the path to a new design methodology called the System on Chip [4].
Instead of using multiple chips to implement a system, a single chip is sufficient to accomplish the same system using System on Chip design methodology.

System on Chip methodology involves the integration of predesigned and already verified blocks called the Intellectual Property (IP) blocks [9] on a single chip. These blocks are reusable [10]. The processing units, memory blocks, input/output interface blocks and other components capable of handling specific applications come under these IP blocks. Besides these, software components which run the applications are also fed into the chip. There are several advantages when a large system constituting multiple chips is integrated on a single chip. They are cost effective, offer high reliability, faster task execution and less power is consumed. Also, because of the integration onto a single chip, the physical size of the system is reduced greatly.

The advances in the semiconductor process technology enabling billions of transistors to be incorporated on a given chip, clock speed limitations and SoC design methodology have led to the emergence of Chip Multiprocessors. The need for faster, parallel and high performance computing, while mitigating the heat dissipation due to increase in the clock frequencies, has been satisfied by CMP. The integration of multiple processing cores onto a single IC die is called as a Chip Multiprocessor. The basic principle behind the CMP architectures to improve performance is by taking advantage of instruction and thread level parallelism. In the current era of high performance, parallel and distributed computing, the advent of CMP has provided solution to most of the problems [11].
The figure 2.4 shows a CMP architecture example with 3 processor cores, each with a register and an ALU. These cores interact via a bus interface.

2.2 Advantages of CMP

Chip Multiprocessors offer several advantages. One of the main advantages is that the latencies because of inter chip communication disappear as the cores are integrated on a single chip. Because of the CMP technology, multiple energy efficient processor cores can be embedded on a single IC die. As work is done parallel, large amounts of data can be processed at high rates increasing the throughput. However there is an upper limit to the number of processors that can be added to a parallel computing platform processing a fixed size program because of the implicit sequential part in that program and
parallelization overhead [13]. As there are multiple processing cores, these can take better advantage of instruction and thread level parallelism. Chip area is efficiently used as the smaller sized less complex cores are embedded on a single chip. Switching between the cores during the application based on the requirements and turning down power to the unused cores increases the efficiency of the system.

2.3 Challenges in Chip Multiprocessors

![Amdahl's Law](image)

Figure 2.5 Speed up vs. Number of processors
The figure 2.5 shows the graph representing Amdahl’s law. Amdahl’s law states that in a computing environment, increasing the parallelism by a number N can never increase the performance by a factor of N [13]. This law gives the relationship between the expected speedup of parallelized implementations of an algorithm relative to serial algorithm. This law assumes that the problem size remains the same even after parallelization [16]. Speed up is defined as the time it takes to execute a program in a serial manner divided by the time it takes to execute that program in parallel.

With the advent of new technologies and increasing complex parallel applications, designers face severe challenges to meet the demands through chip multiprocessing. As the number of processing cores increase on the chip, the cache memory and bus bandwidth available per core decreases because of the relatively limited size of the die, limited cache memory on the chip. The designer has to take care of the growing memory latencies keeping in check the bandwidth requirements but enabling as many cores on the same die for high throughput [11]. Memory hierarchy architecture plays a key role in the performance and scalability of CMP. For high performance and scalability it is necessary to have high speed communication between the caches, a large shared capacity (L2), fast L2 access delay and effective cache sharing [15]. Thread starvation and priority inversion, which affect the utilization of the multicore processor, may occur if L2 cache is not managed effectively [11]. The power dissipation factor becomes more critical as the number of cores is scaled up. Handling the bandwidth required to accommodate for the input/output requests to/from the cores available is also an important criteria.

In single microprocessors and chip-multiprocessors power dissipation issues have limited the increase in the clock frequencies and that is why there is no significant
increase in the clock speeds since 2003. With the advancement in semiconductor process technology, the amount of logic that can be incorporated on a given chip has increased but the designers were not able to exploit the instruction level parallelism in accordance with the available chip logic density. Cache memory was added to the single processor to fill in the available on-chip area but there was no considerable increase in the performance. This led to the evolution of CMPs to get higher performance [2].

2.4 Types of CMP

CMPs can be classified based on the type of processing elements (PEs) present on the chip. In general, there are two types of CMPs, viz. Homogeneous CMP and Heterogeneous CMP. If all the PEs on the CMP are of the same type, then that CMP is referred to as the Homogeneous CMP. If the CMP is comprised of different PEs in the sense of clock frequency, size of cache etc., then that CMP is referred to as Heterogeneous(or asymmetric) CMP. Heterogeneous CMP provides a key advantage of switching between different cores based on the requirement. But the major challenge faced by the heterogeneous CMP is the arrangement of cores and the storage blocks in the available on-chip real estate. In this thesis, we consider homogeneous CMP which implies that all the PEs present on the CMP are identical.
CHAPTER 3

PROCESSOR MANAGEMENT SYSTEM

3.1 Processor Management in Multi-core Processors

In Multi-core processor system, understanding and managing the cores effectively is crucial for system optimization and performance. In the case where the number of cores on a CMP is small, the system optimization depends on the characteristics of the processing elements and in the case where the number of cores on the CMP is large; it depends on the on-chip interconnects [17] [18]. As the number of cores in a multicore system increase, the inter-connects are prone to cross talk, power supply noise, radiation related problems and time delays [18]. Effective processor management system is essential to avoid these defects as much as possible. As the number of hosts to a given bus increases, the bus arbitration delay increases. Also, as the hosts share the bus in a time division manner, the bandwidth decreases as the number of hosts increase [1]. To satisfy such scalable bandwidth requirement, Network On-chip (NoC) methodology was introduced. The figure 3.1 shows an illustration of a 4x4 grid structured NoC.
In the figure 3.1, a grid of routers connected by the communication links can be seen along with the other fundamental components of a NoC. The fundamental components of a NoC are Network adapters- which act as the interface between the cores and the NoC, routing nodes- execute routing, links-which connect the nodes [19].

Besides a good on-chip interconnection network, using the cores effectively is very important for performance in multicore processor systems. An efficient task assignment is necessary to make use of the cores effectively. Task assignment refers to assigning tasks to the available cores in multicore processor system [21]. A job is a collection of tasks. All the cores present on a chip can be engaged in processing the whole job in parallel or individual cores can process different jobs [1]. The designer has to come up with an effective processor management system that can dynamically choose from these two ways of job execution. Processor management system is an integral part of the OS.
Processor management system of the OS deals with job scheduling and processor allocation [22]. Job scheduling, handled by a JS, refers to arranging the jobs in a queue fashion based on certain criteria. This criterion is based on the type of the scheduling algorithm employed. Thus a JS determines the job that is to be processed next. The job scheduler has to be designed in such a manner that the jobs in the job queue should take maximum advantage of the available resources for optimal performance. Processor allocation deals with the selection of set of processors to service the incoming job from the job scheduler. This is handled by PA.

![Figure 3.2 Processor Management System](image)

The figure 3.2 shows the processor management system with its two components:

- Job Scheduler (JS)
- Processor Allocator (PA)
This thesis concentrates on the PA part of the processor management system in the CMP where the processing elements are connected in mesh topology.

3.2 Processor Allocation Algorithms

CMP is the integration of multiple processing elements on a single integrated chip. Each PE node consists of a processing core and a cache memory [1]. Processor management system plays a key role in a CMP, handling the job scheduling and the processor allocation [22]. Once the job is scheduled to be serviced by the JS, PA selects a set of PEs with a goal to maximize the throughput over a stream of jobs. This set of PEs is selected based on the allocation algorithm used. Allocation algorithms can be categorized into contiguous and non-contiguous. If the PA selects a set of PEs which are physically adjacent and has the same topology as the network topology connecting the processing elements then the allocation strategy employed is referred to as Contiguous allocation. If the allocation strategy selects the PEs without the physical adjacency criterion then it is referred to as Non-contiguous allocation [24] [25]. As a result the incoming job does not need to wait till a set of physically adjacent processing elements become free; it can just be executed on a sufficient number of multiple disjoint processing elements.

Contiguous allocation scheme leads to fragmentation issues and high allocation overheads which degrade the performance significantly as the processing resources are not optimally utilized. Fragmentation can be classified as internal and external fragmentation. Contiguous allocation leads to external fragmentation when there are
enough free processors to execute a job but they are not selected due to the lack of physical adjacency or lack of having the same topology as the processing elements connected in the network. This requirement of physical adjacency and possessing the same topology as that of the network is in short called as Sub-grid recognition. An allocation algorithm is said to have complete sub-grid recognition ability if it is always able to find a sub-grid, if available, for the incoming job [30]. This is a critical feature of a processor allocation algorithm. If the number of processors allocated is more than requested by the job for execution, it is referred to as internal fragmentation. However contiguous allocation offers advantages like minimum communication costs and avoiding bandwidth contention and thus it has been used in most of commercial parallel systems. There are simulations which show that the loss caused due to ineffective processing resources utilization is compensated in case of complete contiguous allocation if fragmentation causes at least a factor of two slowdown [27].

Non-contiguous allocation offers advantage over contiguous allocation in that it eliminates fragmentation and has low allocation and de-allocation overheads. Also this allocation scheme offers fault tolerance and supports adaptive allocation strategies [28]. However the message passing takes more links which may lead to message contention and thus causing communication interference with the other jobs. Even with the disadvantage of message contention, non-contiguous allocation offers a better overall performance compared to contiguous allocation [29].

Most of the present day highly parallel commercial systems are confined to contiguous allocation schemes. The contiguity constraint on the selection of processors results in significant drop in the performance. Hence there is need for allocation schemes
with complete sub-grid recognition ability and low allocation and de-allocation overheads. Designers should come up with allocation strategies that absorb the advantages of both contiguous and non-contiguous allocation, thus resolving fragmentation and message contention problem and improving the utilization of processing resources. This allocation strategy employed in this thesis is a contiguous allocation scheme [1].

An efficient processor allocation scheme is necessary to acquire high performance in CMPs. Several contiguous and non-contiguous allocation schemes have been designed in the quest to find an algorithm with low overheads and high resource utilization. These algorithms vary in complexity and their ability of sub-mesh recognition. Sub-mesh recognition ability is an important characteristic of an allocation algorithm. Sub-mesh recognition ability refers to the ability of the allocation scheme to identify available sub-meshes for the incoming job. The allocation strategy with greater sub-mesh recognition ability delivers greater performance [23]. Most of the allocation algorithms have neglected the allocation and de-allocation overheads but their significance depends on the ratio of overhead to the job execution time [23]. The topology of the network connecting the PEs in a CMP can be exploited to design an allocation strategy specific to that topology and thus increasing the efficiency.

3.3 Job scheduling and Processor Allocation

JS and PA play a major role in improving the performance of a CMP. JS determines the next job that is to be serviced and the PA allots the set of PEs for executing the job.
The selection of the job by the JS is based on the scheduling policy employed. If the PA is not able to find the required number of processing elements for that job execution then the JS takes necessary steps to handle the job based on the scheduling policy. In this thesis, it is assumed that the JS employs FCFS scheduling policy. The limitation of FCFS scheduling policy is in the case where a larger job is waiting to be serviced and the PA could not find required number of free processors and the smaller job following that larger job has to wait for an indefinite amount of time even if there are enough free processors to service the smaller job. This is the blocking nature of the FCFS policy. In a CMP, the incoming job is specified the number of processing elements required for its service [30]. If a PE is already occupied by one of the tasks of a job then it is not released till that particular task is serviced. It implies that at any instant, a busy PE will be processing a task of one job only. This is to make sure that there is no overlapping of the jobs [30]. An efficient PA should make maximum use of the processing resources, have less allocation and de-allocation overheads and should be able to suffice without bottlenecks even when the number of nodes becomes large. The design of a PA is becoming complex in trying to acquire these characteristics.

3.4 Network on Chip (NoC)

Network on Chip offers an assuring and reusable architectural design for the CMPs. It has a 4-layer model unlike the 7-layer OSI reference model [31]. The four protocol layers are the physical layer-deals with the number and physical length of the wires connecting resources and switches, data link layer-deals with the transmission of cell between two
switches or between a switch and a resource, network layer defines the packet transmission over the network and finally the transport layer. NoC architectures offer packet switched communication between the cores in a CMP [32].

Figure 3.3 shows the most popular NoC architecture, the two dimensional mesh network (2D mesh). This topology is used in this thesis.

![Figure 3.3 A 7X7 2D Mesh NoC](image)

In the figure 3.3, R represents the router and PE represents the processing elements which are connected by the network channels as shown, thus forming a mesh topology.
A 2D mesh NoC has a $w \times h$ size mesh of switches along with the resources which are present in the slots as seen in the figure 3.4. These slots are created by the interconnections between the switches [32]. In our case, the switches are routers and the resources are PEs. Each router is connected to all the immediate neighboring routers and one PE. NoC becomes the basis for communication between the PEs in a CMP. PEs can be seen as the nodes connected in a mesh topology. The PEs and the switches communicate by sending messages over the channels connecting them. A channel has two one directional point to point buses. As the number of PEs increase, required bandwidth for communication also increases. To avoid any congestion that may be caused, the routers have buffers associated with them [33]. Mesh topology for CMP has
been considered in this thesis due to its regularity, simplicity and suitability for VLSI implementation [23].
CHAPTER 4

PROCESSOR ALLOCATION

4.1 Processor Allocation Strategy Used

Several contiguous and non-contiguous allocation strategies have been proposed recently for a mesh-connected CMP. Mesh topologies are simple, regular and highly suitable for VLSI implementation [23]. Allocation algorithms such as First Fit [37], Best Fit [37], Improved First Fit, Better First Fit [34], Improved Better First Fit [34], 2D Buddy System [35], Adaptive Scan [38], Frame Sliding [36], Adjacency Strategy [39] etc., have been proposed to allocate the processing elements to the incoming jobs in a CMP with mesh NoC.

4.1.1 K-ary 2-mesh

The following definitions are necessary in order to understand the allocation strategies in a K-ary 2-mesh. A k-ary 2-mesh can be represented by M(w,h) where ‘w’ represents the width of the 2D mesh and ‘h’ represents the height of the 2D mesh as shown in the figure 4.1.
The mesh has \( w \times h \) number of nodes, each node referring a PE. The busy array corresponding to \( M(w,h) \) is \( B[w,h] \) where \( B[i,j] \) is the element with value “0” if the node is free and “1” if the node is busy. \( B[i,j] \) is the element in the \( i^{th} \) column and \( j^{th} \) row. The bottom-most row is given a row number 0 and the left-most column is given the column number 0 as shown in the figure 4.2. These nodes are connected by means of network channels. Each node except for the boundary nodes is connected to four immediate neighboring nodes. All the boundary nodes except for the nodes at the corners are connected to three immediate neighboring nodes. The nodes at the corner are connected.
to two immediate neighboring nodes. All the connections between the nodes can be verified from the figure 4.2.

Figure 4.2 Mesh M(8,8) with row numbers and column numbers

Below are the definitions which help in understanding the allocations strategies better.

Definition .1: A sub-mesh S (p,q) is a sub-grid in the mesh M(w,h) such that $1 \leq p \leq w$ and $1 \leq q \leq h$. A sub-mesh is defined by a base and an end. A base is the bottom left node and end is top right node in a sub-mesh: S (<base coordinates><end coordinates>).
Definition 2: An incoming job of size $p \times q$, which is requesting for a sub-mesh of size $p \times q$ (sub-mesh since it is a contiguous allocation in this thesis), is denoted as $J(p,q)$.

Definition 3: A sub-mesh is said to be a busy sub-mesh $\beta$ if all its nodes are busy which means that all its nodes are allocated to the jobs.

Definition 4: A bitmap $B[w,h]$ of a mesh $M(w,h)$ is an array of size $w \times h$ with elements $B[i,j]$ having values either 1 or 0 depending on whether the node at that position is busy or free respectively.
Definition 5: The base node with respect to an incoming job of size $p \times q$ in a mesh is the bottom leftmost node of a free sub-mesh of size $p \times q$ available in the mesh for that job allocation.

Definition 6: Reject area $R_J$ of a mesh with respect to a job $J$, is defined as the set of nodes such that any node from that set selected as a base node for the incoming job $J$ will result in the job to cross the boundary of the mesh. Thus none of the nodes from the reject area are eligible for job allocation.

Figure 4.5 shows the reject area with respect to a job $J(2,3)$ marked with the box region.

\[
B[6,7] = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Figure 4.4 Bitmap showing the status (busy/free) of the processors
Definition 7: The sink of the reject area is the node with coordinates <w-p+1, h-q+1>. The sink helps in marking the reject area easily.

Definition 8: The coverage of a busy sub-mesh $\beta$ with respect to an incoming job $J$ is the set of nodes (or processors) which when chosen as the base for the job $J$ will result in the overlapping of the job over the busy sub-mesh $\beta$. The coverage of a busy sub-mesh $\beta$ with respect to the job $J$ is denoted as $\xi_{\beta,J}$. The set of all such coverages with respect to the job $J$ is defined as Coverage Set $C_J$. 

Figure 4.5 Reject area with respect to the job J(3,2)
4.1.2 Bitmap Approach

Improved First Fit allocation algorithm has been adopted by the PA in this thesis since it delivers a good performance among the algorithms based on the bitmap [1]. As the name suggests the Improved First Fit algorithm has been developed from the First Fit algorithm. The First Fit and the Best Fit algorithms are based on the Bitmap approach [37]. Bitmap approach is also called as Busy Array approach. In the bitmap approach, the status of the processors is represented in the form of a bitmap. If the processor in the mesh is free it is given a value “0” in the bitmap and if it is busy then it is given a value “1” in the bitmap. An example showing a bitmap representing the free and busy processors can be seen in the figure 4.7.
In the bitmap approach, a busy array $B$ is created based on the status of the processors connected in the mesh. This busy array $B$ is stripped off the reject area $R_I$ and then it is scanned to obtain the coverage array $C_T$. This results in a bitmap with the coverage set. To efficiently form the coverage array $C_T$, each coverage is divided into smaller coverages, viz., the job coverage, the left coverage and the bottom coverage. These coverages can be seen in the figure 4.8.
To determine job coverage, left coverage and bottom coverage the following two scans are required.

- **Scan 1**: Scanning the rows from right to left to determine the job and left coverages.
- **Scan 2**: Scanning each column from the top to the bottom starting from the extreme left column and ending at the extreme right column.

Both the First fit and the Best fit allocation strategies are identical till the above the scanning process. The FF algorithm returns the first available free node in the bitmap (without the reject area) obtained after the formation of coverage set. The Best fit algorithm returns that free node as the base which has maximum number of busy neighbors. Thus the Best fit algorithm requires an additional scan to determine the base
for the incoming job whereas in case of the First fit algorithm, the scanning is interrupted whenever the first free node is encountered in the bit map. Both First fit and Best fit are contiguous allocation schemes. External fragmentation and system utilization are almost the same in both FF and BF allocation strategies. It has been observed that the FF strategy performs better than the BF strategy because of the scattering of new jobs in BF allocation which occurs due to the dynamic nature of the incoming jobs [37]. Hence IFF allocation strategy which is based on the FF strategy has been used in this thesis.

4.1.3 First Fit Algorithm and Improved First Fit Algorithm

The First fit allocation algorithm follows these steps to find the base coordinates for the incoming job.

Step 1: Calculate the coordinates of the sink with respect to the incoming job J in the bitmap B of the mesh M.

Step 2: for each row in B do

for each node in considered row from right to left do

- create job and left coverages in $C_T$ with respect to J

Step 3: for each column from left to right in $C_T$ without $R_J$ do

for each node in considered column from top to bottom do

- create bottom coverage in $C_T$ with respect to J
if node can be base then

    return node as base for J (success)

Exit

Step 4: return fail (job is not allocated)

Exit

The time complexity during allocation using FF strategy is $O(wh)$. It can be observed in the above algorithm that the FF strategy considers only a fixed orientation of the incoming job which results in the lack of recognition completeness. Lack of recognition completeness may result in the job not being allocated even when a sufficient sub-mesh of free processors is available. The following explains this scenario [1]. Consider a case where there is a free sub-mesh of size $w \times h$ such that $w \neq h$ available for the incoming job of size $p \times q$ ($p \neq q$) and let $p=h$ and $q=w$. The number of available processors for allocation is $w \times h$ and the number of free processors required for the incoming job is $p \times q$. $p \times q = w \times h$ since $p=h$ and $q=w$, but the processors would not be allocated to the incoming job because of the mismatch in corresponding heights and widths of the job and the free sub-mesh. This is due to the fixed orientation of the incoming job and thus the algorithm lacks recognition completeness.

The First fit algorithm has been improvised to acquire recognition completeness. The new algorithm, called the Improved First Fit algorithm, employs the same strategy as the FF algorithm with an additional step. The additional step involves the change in the
orientation of the job in case the algorithm fails to allocate the processors in the original orientation. The IFF strategy employs the following algorithm.

Step 1: Calculate the coordinates of the sink with respect to the incoming job J in the bitmap B of the mesh M

Step 2: for each row in B do

for each node in considered row from right to left do

• create job and left coverages in \( C_T \) with respect to J

Step 3: for each column from left to right in \( C_T \) without \( R_J \) do

for each node in considered column from top to bottom do

• create bottom coverage in \( C_T \) with respect to J

if node can be base then

return node as base for J (success)

Exit

Step 4: if the orientation of the job was not changed and \( p \neq q \) then

• \( J(p,q) \leftarrow J(q,p) \)

• Go to step 1

else
As it can be noticed from the above algorithm, the orientation of the job is changed in the case the algorithm fails to allocate the processors for the job in its original orientation. Considering both the orientations of the job assures complete sub-mesh recognition [1]. But this adds an additional overhead compared to the schemes that deal with only one orientation of the job. Each time the PA is called to allocate a job, it results in an allocation overhead. Minimizing the allocation overhead is critical to the processor allocation algorithms. In the case where an allocation algorithm with complete sub-mesh recognition is used to allocate a job $J(p,q)$ and $p=q$, the scheme can be employed for just one orientation of the job to guarantee complete sub-mesh recognition [1]. This eliminates the redundant runtime and minimizes the allocation overhead as the algorithm is employed just once. IFF offers good utilization and performance for mesh networks and hence it has been considered in this thesis [1].

4.2 Processor Allocator Architecture

Architecture of the PA plays a key role in Chip multiprocessors. Using on-chip real estate efficiently is crucial to the CMP design. Thus there is a need to come up with a PA architecture which is area and energy efficient. PA is one of the two key components of the processor management system of the OS. PA allocates a set of processors to the
incoming job based on the allocation scheme used. The architecture of the PA depends on
the allocation algorithm employed by the PA. The PA architecture based on the bitmap
approach [40] can be seen in the figure 4.9.

![PA Architecture Diagram](image)

**Figure 4.9** The PA architecture based on the bitmap approach [40]

In the figure 4.9, “clk” is the clock signal used to trigger the PA which is the same
clock signal given to the Job scheduler (JS). \( J_A \) is the job that is to be allocated by the PA.
\( J_D \) represents the job that is to be deallocated and \( J_O \) represents the output which has the
job number and the corresponding base coordinates found by the PA. Each job \( J \) is
associated by a job number, base coordinates and its size.

\[ J<\text{job number}><x_b><y_b><p><q> \]

The job number is like an ID tag which is unique to each job and is issued by the OS.
\( x_b, y_b \) are the base coordinates which are calculated by the PA based on the allocation
algorithm employed. \( p \) and \( q \) together denote the size of that particular job. The incoming
job \( J_A \) which is to be allocated is associated with a job number given by the OS, its size
represented by $p$, $q$ and empty values for $x_b$, $y_b$. Once the PA calculates the base coordinates for that particular job, $J_0$ with the same job number and size as the corresponding $J_A$ in addition to the values for $x_b$, $y_b$ is sent out as the output. If a job is serviced and its de-allocation is requested, the $J_0$ which has the information about the job number, size and corresponding base coordinates is given as the input through $J_D$ and necessary action for deallocation is taken. “ack” is the acknowledgment output from the PA which conveys if the job is allocated or not. It is a two bit output representing the status of allocation. If the output at $ack$ is “00” or “11” then it signifies that the job is allocated by the PA. If the output at $ack$ is “01” or “10” then it means that the job is not allocated by the PA. The width was chosen to be two bits for $ack$ to accommodate for any extensions in the future [40].

4.2.1 Processor Allocator Architecture Used

The architecture of the PA used in this thesis is drawn from the PA architecture proposed in [1]. The size of the incoming job and a clock signal $clk$ is given as the input to the processor allocator. The reset button $rst$ is also given as the input to reset the PA after the result for the incoming job is obtained. The outputs $b_x$ and $b_y$ denote the base coordinates for the incoming job. IFF algorithm is implemented by this processor allocator to find the base coordinates for the incoming job. The output $ack$ is an acknowledgement signal which gives the status of the PA. The width of the $ack$ signal can be modified based on the number of states of the PA.
Figure 4.10 The Processor Allocator Model used in this thesis
CHAPTER 5

HARDWARE IMPLEMENTATION

The objective of this thesis is the hardware implementation of the processor allocator. PA allocates a set of processors to the incoming job based on the allocation algorithm employed. In this thesis, the Improved First Fit algorithm is used for processor allocation in a mesh connected CMP. The implementation procedure offers a basic structure to implement even the other allocation strategies. The hardware implementation has been done on Virtex 4 ML403 Evaluation Platform and Xilinx Spartan 3E Starter Kit. This chapter explains the whole process from the design entry into the Xilinx ISE 13.2, analysis of various reports and the bit file generation. This bit file is then programmed into the FPGA and implemented.

Current FPGAs offer several advantages like high performance, fault tolerance due to their dynamic reconfiguration capability. A VHDL code has been written executing the IFF algorithm and thus finding a set of free processors (if available) in a mesh connected CMP. Simulation and synthesis results for various mesh sizes and job sizes have been performed. The synthesis and various analysis reports provided in this chapter are for a mesh M(6,6). The steps below give an idea about the way the code has been written in VHDL for hardware implementation.
The VHDL code is written following these steps.

Step 1:

The Processor Allocator is in constant interaction with the mesh connected processors of the CMP. When the PA encounters an incoming job, a bitmap is created based on the current status of the processors in the mesh.

Step 2:

Using the size parameters of the incoming job as inputs, reject area and coverage areas are found for the bitmap. The nodes/processors present in these regions are marked ineligible for being a base to the incoming job.

Step 3:

The coordinates for the first encountered free node which can act as a base node for the incoming job are recorded and an acknowledgement is sent to the PA reporting a “success” and then the processors are allocated to the incoming job accordingly and PA moves on to the allocation of the next incoming job.

Step 4:

If a base node could not be found after step 3 then an acknowledgement denoting a “failure” is sent to the PA.

Step 5:

In case of a failure in step 5, the orientation of that job is changed and processed through steps 2 to 4. If there is a “failure” even after change in orientation, this
information is sent to the PA acknowledging a “failure after swapping” and the PA moves on to the next incoming job.

The handling of the job in case of a failure in both the orientations is not dealt in this thesis. The job can be put on hold till the required sub-mesh of processors are free or the job can be dropped.

Figure 5.1 shows the block diagram of processor allocator module used in this thesis as an input-output system.

![Diagram of Processor Allocator module](image)

**Figure 5.1** The Processor Allocator module used for Hardware implementation

In figure 5.1, $p$, $q$, $clk$, $rst$ and IFF are the inputs to the PA module. $b_x$, $b_y$ and $ack$ are the outputs from the PA module. $p,q$ denote the size of the incoming job, $clk$ is the system
clock used and $rst$ is the system reset. IFF is the allocation strategy employed by the PA module. $b_x$, $b_y$ represent the base coordinates found for the incoming job using IFF allocation strategy. $ack$ is the acknowledgement pin which specifies whether a base is found for the incoming job or not. The width of the $ack$ pin depends on the amount of details the user wants regarding the status of processor allocation.

5.1 Implementation on ML403 Evaluation Platform

The board used to achieve the objective of this thesis is a Virtex 4 ML403 Evaluation board. This board is powered by virtex-4 XC4VFX12 device and has peripherals, interfaces and connectors which are compliant to industry standards. The FPGA has a speed grade of -10 and a FF668 package.

The board has two clock sockets, one with a 100MHz oscillator and the other with a 3.3V clock oscillator. It has a 16X2 character LCD as a display, push buttons and LEDs as general purpose input/output and a 64-bit user expansion connector. The ML403 board is not equipped with DIP switches. 64MB DDR SDRAM, 8Mb ZBT SRAM, 64Mb Flash, 4Kb IIC EEPROM comprise the memory of the ML403 board. The board also has one differential clock input pair and one differential clock output pair with SMA connectors. The board is also equipped with a JTAG slot, RS232 port, PS/2 mouse and keyboard connectors and other USB interfaces. The board has a 5V @ 3A AC adapter for power supply [41].
The software tool used is the ISE Design Suite 13.2 from Xilinx. A VHDL code for the PA using IFF algorithm is written to create the system module in Xilinx ISE Design Suite 13.2. Test-benches are written and simulations have been done on the Isim simulator to verify the outputs for the given set of test inputs.

VHSIC Hardware Description Language (VHDL) is not a programming language but a hardware description language for integrated circuit design. The system level
components and non-synthesizable test benches can be described easily with VHDL because of its high levels of abstraction. But many VHDL constructs are not supported by the synthesis tools. Hence the designer must make sure to use a synthesizable construct while writing the VHDL code. Here the VHDL code is synthesized using Xilinx ISE13.2 and thus the VHDL constructs which are synthesizable in the Xilinx environment have to be used [42]. VHDL is used since it provides great flexibility in describing the hardware.

The figure 5.3 to figure 5.8 show how the design is entered using VHDL in Xilinx ISE Design Suite 13.2 and various reports are generated and analyzed and finally generating a bitstream which is implemented on the board. Thus hardware implementation of the processor allocator for a mesh connected CMP is done.
Figure 5.3 Device Utilization Summary using a ML403 Evaluation Platform
Figure 5.5 Screenshot showing detailed reports in ML403 Evaluation platform
Figure 5.6 Package while implementing on ML403 Evaluation Platform
Figure 5.7 Place and Route completion in ML403 Evaluation Platform
Figure 5.8 Screenshot showing the Bitstream generation in ML403 board
The generated bitstream is then loaded into the ML403 Evaluation Platform and thus hardware implementation of the processor allocator is performed.

5.2 Implementation on Spartan 3E starter Kit

The board used for hardware implementation is the Spartan 3E FPGA starter kit. The board is equipped with a powerful Xilinx XC3S500E FPGA which has 320 pin FBGA package and has over 10,000 logic cells. Some of the key features of this board include Xilinx 4 Mbit Platform Flash Configuration PROM, 16 MByte of parallel NOR flash, 16Mbits of SPI serial Flash, 16X2 LCD display, VGA display port, On-board USB based FPGA/CPLA download/debug interface. It is also equipped with a 50 MHz clock oscillator, eight discrete LEDs, four slide switches and four push-button switches. The figure 5.9 shows the Spartan 3E FPGA starter kit.
The software tool used for synthesis and analysis of HDL designs in this thesis is Xilinx ISE13.2. The PA module is written in VHDL as it offers great flexibility to hardware description. Many VHDL constructs are not supported by the synthesis tools. Hence the designer must make sure to use a synthesizable construct while writing the VHDL code. Here the VHDL code is synthesized using Xilinx ISE13.2 and thus the VHDL constructs which are synthesizable in the Xilinx environment have to be used.
A test bench is written and the code is checked for functional correctness for several test inputs. A debouncing module is added to the top module to negate the bounce from the toggle switches and push-button switches present on the board. Simulations are performed on using Isim simulator. XST is used to synthesize the design. A synthesis report showing the logic utilization is generated.

The following screenshots show how the design is entered into Xilinx ISE13.2 and various reports are generated. The reports are analyzed using various tools available in the Xilinx ISE13.2 and a bit file is then generated and programmed into the board. Thus the hardware implementation of Processor allocator for mesh-connected CMP is done.
Figure 5.10 Device Utilization Summary using a Xilinx Spartan 3E Starter Kit
Figure 5.11 Package schematic while implementing on Xilinx Spartan 3E Starter Kit
Figure 5.12 Design schematic while implementing on Xilinx Spartan 3E Starter Kit
Figure 5.13 Clock Net while implementing on Xilinx Spartan 3E Starter Kit
Figure 5.14 Completion of Place and Route for Xilinx Spartan 3E Starter Kit
Figure 5.15 Completion of Bitstream for Xilinx Spartan 3E Starter Kit
The generated Bitstream is programmed into the board and then implemented on the board. Thus the hardware implementation of the processor allocator for mesh-connected CMPs is done.
There has been an enormous research and development in the field of parallel computing. Parallel computing is the solution to many modern day computing problems offering efficient performance. The number of transistors on a single IC has grown exponentially from couple of thousands to billions in the past forty years due to the advancement in the semiconductor process technology. This is in accordance to the Moore’s law. The increasing transistor density, complexity of the core and processor design has led to the emergence of multi-core architectures to boost the performance. Multicores provide better energy efficiency and higher clock rates as compared to the large single core design. The need for faster, parallel and high performance computing, while mitigating the heat dissipation due to increase in the clock frequencies, has been satisfied by CMP. The basic principle behind the CMP architectures to improve performance is by taking advantage of instruction and thread level parallelism. In the current era of high performance, parallel and distributed computing, the advent of CMP has provided solution to most of the problems. The inter-core communication latency is very low as the cores reside on a single chip in a CMP. As the modern CMPs use NoC, this increases the performance in terms of communication when compared to the bus architectures connecting the cores. Mesh networks are well suitable to many practical applications like image processing and matrix computations and hence this network topology has been considered in this thesis.
The operating system plays a crucial role in handling the jobs that are to be processed by the processing elements connected by a network in a CMP. The processor management system of the OS deals with aligning the jobs in order of execution and in allocating the processors required for servicing the job. These two operations are done by the Job Scheduler and the Processor allocator components of the processor management system respectively based on the algorithms employed by them.

This thesis deals with PA component of the processor management system. Here, the PA is implemented on the hardware. The processing elements are assumed to be connected in a mesh topology. An Improved First Fit algorithm which is a contiguous allocation strategy has been used for allocating processing elements by the PA to the incoming jobs. The hardware implementation has been done on Virtex 4 FX12 Evaluation board and Xilinx Spartan 3E starter kit.

The PA design entry has been done by writing a VHDL code and then necessary simulations have been done to check the functionality of the design. Then it is synthesized and implemented on Virtex 4 FX12 Evaluation board and Xilinx Spartan 3E starter kit. The synthesis reports and various analysis reports are presented in this thesis.

The VHDL code used in this thesis can be modified according to the topology used by the processors and the allocation algorithms used with slight adjustments in the code. Thus this acts as a basis for the hardware implementation of most of the contiguous allocation strategies. The thesis also provides a brief understanding of the background of the CMPs. The PA architecture has been explained and synthesized. The synthesis report confirms good results of the PA using an IFF allocation strategy.
6.1 Future Scope

There is a need for an effective PA to get an optimal performance in a CMP. The architecture of a PA can be improved a lot based on the specifications needed for that particular job allocation. The real estate occupied by the PA cannot be reduced significantly. However developments can be made to improve the operational frequency of the PA and thus increasing its performance. CMP with an integrated PA and JS has to be designed to mitigate the communication latency between the network of cores on the CMP and the PA. An efficient allocation solution for higher dimension networks is an important area of research. The hardware implementation of the PA using the softcores such as microblaze in the Xilinx boards would yield better results. One of the future scopes can be an ASIC implementation of the CMP as in this case the computational power of the processing cores could be much higher compared to that of FPGA board. There is a need for efficient allocation algorithms which can eliminate the fragmentation issue, minimize allocation overhead, reduce job turnaround times and maximize the system utilization [43].

Different allocation strategies offer different advantages depending on the parameters at the time of allocation of the incoming job. Designing a single PA that can implement a specific allocation strategy out of a set of allocation strategies to take the maximum advantage of the scenario present at the time of processor allocation can yield a very high performance. This is because of the fact that a specific algorithm can be advantageous in a specific scenario when compared to most of the allocation strategies. This thesis deals with only the allocation strategy of the processor allocator but not the de-allocation strategy. Once a job is serviced by the processors, the processors allocated to that job
should be made available to the other incoming jobs. This process of de-allocation along with the allocation strategy can be implemented in the hardware making it a dynamic system dealing continuously with the allocations of the incoming job and de-allocation for the serviced jobs.


[Online]. Available:


VITA

Graduate College
University of Nevada, Las Vegas

Rana Sangram Reddy Marri

Degrees:

Bachelor of Technology in Electronics and Communication Engineering, 2009
Jawaharlal Nehru Technological University, India

Thesis Title:
Hardware Implementation of Processor Allocator for Mesh Connected Chip Multiprocessors

Thesis Examination Committee:
Chairperson, Dr. Henry Selvaraj
Committee member, Dr. Emma Regentova
Committee member, Dr. Yingtao Jiang
Committee member, Dr. Laxmi P. Gewali