A SURVEY OF PARADIGMS FOR BUILDING AND DESIGNING PARALLEL COMPUTING MACHINES

Ahmed Faraz [1] Faiz Ul Haque Zeya [2] and Majid Kaleem [3]

[123]Department of Computer and Software Engineering, Bahria University Karachi Campus, Stadium Road Karachi, Pakistan

ABSTRACT

In this paper we describe paradigms for building and designing parallel computing machines. Firstly we elaborate the uniqueness of MIMD model for the execution of diverse applications. Then we compare the General Purpose Architecture of Parallel Computers with Special Purpose Architecture of Parallel Computers in terms of cost, throughput and efficiency. Then we describe how Parallel Computer Architecture employs parallelism and concurrency through pipelining. Since Pipelining improves the performance of a machine by dividing an instruction into a number of stages, therefore we describe how the performance of a vector processor is enhanced by employing multi pipelining among its processing elements. Also we have elaborated the RISC architecture and Pipelining in RISC machines After comparing RISC computers with CISC computers we observe that although the high speed of RISC computers is very desirable but the significance of speed of a computer is dependent on implementation strategies. Only CPU clock speed is not the only parameter to move the system software from CISC to RISC computers but the other parameters should also be considered like instruction size or format, addressing modes, complexity of instructions and machine cycles required by instructions. Considering all parameters will give performance gain . We discuss Multiprocessor and Data Flow Machines in a concise manner. Then we discuss three SIMD (Single Instruction stream Multiple Data stream) machines which are DEC/MasPar MP-1, Systolic Processors and Wavefront array Processors. The DEC/MasPar MP-1 is a massively parallel SIMD array processor. A wide variety of number representations and arithmetic systems for computers can be implemented easily on the DEC/MasPar MP-1 system. The principal advantages of using such 64×64 SIMD array of 4-bit processors for the implementation of a computer arithmetic laboratory arise out of its flexibility. After comparison of Systolic Processors with Wave front Processors we found that both of the Systolic Processors and Wave front Processors are fast and implemented in VLSI. The major drawback of Systolic Processors is the problem of availability of inputs when clock ticks because of propagation delays in connection buses. The Wave front Processors combine the Systolic Processor architecture with Data Flow machine architecture. Although the Wave front processors use asynchronous data flow computing structure, the timing in the interconnection buses, at input and at output is not problematic..

KEYWORDS

Pipelining, Processing Elements, Vectors, Vector Processors, MIMD machine, SIMD machine, DEC/MasPar MP-1 system, Systolic Array Processors, Wave front Array Processors.

1. INTRODUCTION

The essential characteristic of any parallel algorithm is that it can be executed on MIMD model of parallel computers in an efficient manner. Due to this reason MIMD model is used for building parallel computers for diverse applications. Since MIMD parallel computers are equally capable of running a wide variety of applications therefore such computers are said to have General Purpose Architecture. Also we have a second category or type of Parallel Computers Architecture

DOI : 10.5121/cseij.2015.5101

in which the parallel computer design is problem specific. That is we have to assemble various processors in a designed configuration specifically for the problem under consideration. Obviously such type of architecture is very costly because of problem specific design. The outcome is that such type of parallel computer solves a particular problem in an efficient and timely manner. These parallel computers cannot be used for any other type of problem solving and for any other purpose. Such a parallel computer is said to have a special purpose Architecture. In order to design and build parallel computers, there is a diverse range of paradigms; they all fall into several general categories. First of all we introduce and elaborate the concept of Pipelining then we describe the general categories of Parallel Computers:

2. UTILIZATION OF PIPELINING IN PARALLEL MACHINES

In order to review the Parallel Computing Machines first we elaborate the concept of Pipelining and we emphasized the utilization of Pipelining in the design and architecture of different Parallel Computing Machines.

2.1. Pipelining

The Computer Architecture employs parallelism and concurrency through Pipelining. Pipelining strategy can be understood by making a comparison between Pipelining and assembly line in an industrial manufacturing plant. An assembly line takes advantage of the fact that a product goes through various stages of production. By laying the production process out in an assembly line, products at various stages can be worked on simultaneously. This process is called Pipelining [2]. The concept of Pipelining is that new inputs are accepted at one end before previously accepted inputs appear as outputs at the other end. We can apply the concept of Pipelining to the "Instruction Execution". An Instruction can be divided into a number of stages. Generally an Instruction Cycle can be divided into ten tasks which occur in a particular order. Therefore we can state that an Instruction Cycle can employ Pipelining. Let us consider subdivision of Instruction Processing into two stages: Fetch Instruction and Execute Instruction. There are times during the execution of an instruction when main memory is not being accessed. This time can be utilized in performing two tasks□1) Fetching the next instruction (2) Executing the Current Instruction .Therefore the Pipeline has two independent stages. The first stage fetches an instruction and buffers it. Then the first stage waits for the availability of the second stage. When the second stage is free, the first stage passes the buffered instruction to the second stage. While the second stage is executing the instruction, the first stage takes advantage of any unused memory cycles to fetch and buffer the next instruction. This is called "Instruction Pre fetch or Fetch Overlap". The superior advantage of Pipelining is that it speeds up the instruction execution [2]. Let us consider a simple case. A certain computer has an Instruction Pipeline having only two stages: Fetch and Instruction Execute. These two stages have equal duration. Therefore for this case the instruction cycle time would be halved. However for this particular case we have two conditions: 1- The execution time will generally be longer than the fetch time. Execution will involve reading and storing operands and the performance of some operation. Thus the fetch stage may have to wait for some time before it can empty its buffer. 2- A conditional branch instruction makes the address of the next instruction to be fetched unknown. Thus the fetch stage must wait until it receives the next instruction address from the execute stage. The execute stage may then have to wait while the next instruction is fetched. There is a time loss due to second reason. We can reduce the time loss by using guess. There is a simple procedure for it. When a conditional branch instruction is passed on from the fetch to the execute stage, there is a specific order of instruction execution. First the branch instruction is executed in memory then fetch stage fetches the next instruction in memory after branch instruction. The loss of time occurs in the Branch taken. If the branch is not taken, no time is lost. If the branch is taken, the fetched instruction must be discarded and a new instruction is fetched. Because of these factors a two stage pipeline becomes less effective but a smaller gain in speedup occurs. To gain more speedup, the size of Pipeline should be enhanced. That is the Pipeline should have more number of stages.

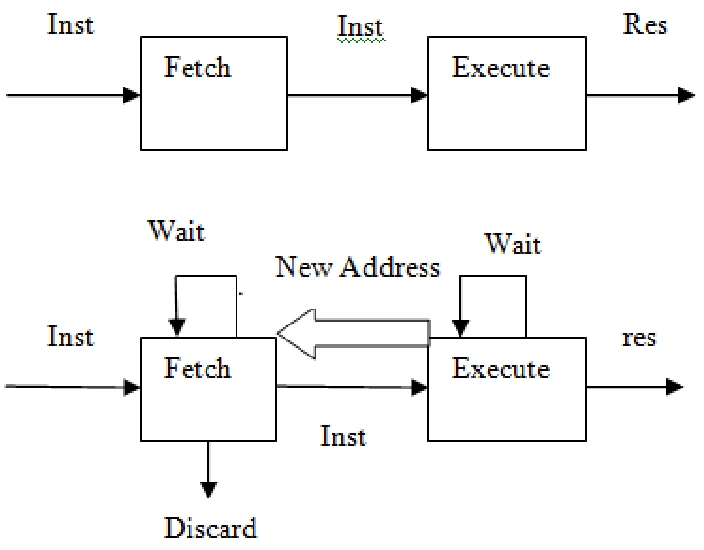

Figure. 1 Instruction Pipelining involving two stages (a) Simplified View (b) Expanded View

2.2. Pipelining Stages

The Instruction Pipelining is decomposed into the following sub stages:

2.2.1. Fetch Instruction (FI)

Read the next expected instruction into the buffer.

2.2.2. Decode Instruction (DI)

Determine the op code specifier and the operand specifier.

2.2.3. Calculate Operands (CO)

Calculate the effective address of each source operand. This involves the calculation of effective address on the basis of type of addressing mode used in the instruction. For example an instruction may use displacement, indirect, register indirect or other forms of address calculation.

2.2.4. Fetch Operands (FO)

Read the memory and fetch the operands. The operands in the registers are not fetched.

2.2.5. Execute Instruction (EI)

Perform the indicated operation which is specified by the op code. And store the results. The results are stored in the specified destination address location.

2.2.6. Write Operand (WO)

Store the result in memory. This is the last stage of Instruction Pipeline which delivers the output.

2.3. Pipelining in Parallel Computing Machines

Suppose that we have many data items D_0, D_1, D_2, D_3, ---$D_{n-1}$and they need to be operated upon at the same time instant‘t’. In an architecture that employs “Pipelining” we have different “Processing Elements” PE_1, PE_2, PE_3, -----PE $_n$. These processing elements are connected to each other in a chain. Data items D_1, D_2, D_3, ---- D_n are passed to Processing Elements PE_1, PE_2, PE_3, ---- PE_n in a sequential manner. Each data item is passed individually and separately to each processing element. There are different classes of computers that employ pipelining. The first one is RISC (Reduced Instruction Set Computer).

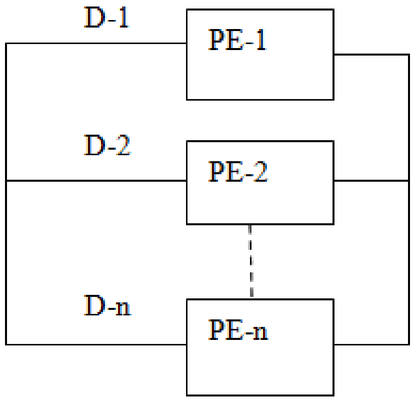

Figure2. Pipelining in Parallel Computing Machines

3. RISC Computers versus CISC Computers

The RISC stands for Reduced Instruction Set Computers. Although there are various characteristics of RISC Architecture but the four principal characteristics of RISC Architecture are following:

- One Instruction per Cycle
- Register to Register Operations
- Simple Address Modes
- Simple Instruction Formats

First we elaborate the concept of Machine Cycle. A Machine Cycle is defined to be the time it takes to fetch two operands from register, perform an ALU operation and store the result in a register. This implies that the RISC machine instruction is not a complex machine instruction as compared to CISC machine instruction. But RISC machine executes about as fast as microinstructions on CISC machines. Since RISC instructions are One-Cycle Instructions therefore there is no need of microcode instructions in RISC. The machine instructions can be hardwired. Such machine instructions run faster as compared to the comparable machine instructions on other machines. When instruction executes it is not mandatory to access the micro program control store during instruction execution on RISC. The second characteristic is that most operations on RISC machines are register to register. To access memory only LOAD and

STORE instructions are used. The instruction set of RISC machine and the design of Control Unit of RISC machines are simplified by using the above mentioned design features of the architecture. Let us mention an example of a RISC instruction. RISC instruction set consists of only two instructions for performing addition operation. The first one is integer add and the second one is add with carry. Now consider a CISC machine: VAX machine. The VAX machine has 25 different ADD instructions. The other feature of RISC architecture is that the "optimization of register use" is the hall mark of RISC architecture. The operands which are most frequently accessed are placed in high speed storage. First we discuss Pipelining in RISC computers then we compare RISC computers with CISC computers with reference to parameters like instruction set, complexity of instructions, addressing modes, instruction format, internal register set, machine cycle required by instructions and CPU clock.

3.1. Pipelining in RISC

The RISC (Reduced Instruction Set Computer) employs Pipelining. The other class of computers that employ Pipelining is Super Scalar Processors. It is manifested that implementation of arithmetic or logical or other type of operations through pipelined architecture improves the performance of computing machines and gives an efficient output with lesser time of execution [7].

Let us consider generally a Parallel Computing Machine. The following symbolic notation represents the data transfer operation of i-th Data Element or Datum to i-th Processing Element.

$DE_1 \longrightarrow PE_1$
$DE_2 \longrightarrow PE_2$
$DE_3 \longrightarrow PE_3$

$DE_n \longrightarrow PE_n$

Here the typical conventions or nomenclature for the above data transfer functions is following:

DE_i: i-th Datum or Data Element
PE i: ith Processing Element

The contents of Data Elements are transferred to the Processing Elements for arithmetic and logical operations. Here DE_n is the nth Data Element and PE_n is the nth Processing Element.

3.2. Instruction Set

The instruction set of CISC computer is larger as compared to RISC computers. The size of CISC computers instruction set is greater than 100 whereas the size of instruction set of RISC computer is within the range of 50-75 instructions.

3.3. Complexity of Instructions

The CISC instructions are more complex as compared to RISC instructions. The CISC instructions are micro coded where as RISC instructions are not micro coded

3.4. Addressing Modes

The addressing modes used in CISC computers are more complex as compared to RISC machines addressing modes. The addressing mode which is most often used in RISC machines is register direct addressing mode or register indirect addressing mode. The RISC instructions usually operate on only register operands.

3.5. Instruction Format

The instruction size in CISC computers is variable whereas the instruction size in RISC computers is fixed
.

3.5. Internal Register Set

The internal register set of CISC computers is smaller whereas the internal register set of RISC computers is larger. The internal register set of CISC computers contain less than and equal to 32 instructions whereas the internal register set of RISC computers contain instructions within the range of 32 - 256 instructions

3.5. Machine Cycle

Multiple machine cycles are required by most of the instructions in CISC instructions set whereas one machine cycle is required by most of the instructions in RISC instructions set.

3.5. CPU Clock

The clock speed of CISC computers is slower as compared to the RISC computers. The CPU clock speed of CISC computers is less than 30 MHz where as the CPU clock speed of RISC computers is within the range of 25-100 MHz.

4. Classification of Parallel Computing Machines

Following is the detailed discussion of different classes of Parallel Computing Machines.

4.1. Array Processors

Array Processors are parallel computer systems having a Single Control Unit. There are a number of processing elements PE_1, PE_2, PE_3, -----, PE_n with nearest neighbour connection. These processing elements operate in parallel under the control of Single Control Unit. The essential characteristic of the Processing Elements in an Array Processor is that each of the Processing Elements performs similar operation at the same time on different Data Elements. This means that if the Processing Element PE-i performs “Addition” operation on different data stream which is D(i) ,D(i+1),D(i+2) then all Processing Elements other than PE(i) will also perform “Addition” operation on different data streams. When we compare SIMD computers with Array Processors, we see that Array Processors are subclass of SIMD computers.

4.2. Vector Processors

Most Vector Machines or Vector Processing Based Computer Systems have pipelined structure. Vector Processors perform computation through “Vectors”. Vector Processors have specially designed registers to hold Vectors. Vector Processors are very powerful machines. These Vector Processors utilize Multi pipelining concept. When one pipeline is not enough to achieve high performance, usually more than one pipe lines are provided. The hallmark of Vector Processors is the parallel operation of several pipe lines on independent stream of data on the same time. In Vector Processors high performance is not only achieved through the use of pipe lining but it is also achieved by the parallel operation of multiple pipelines on independent stream of data [7]. When we compare Vector Processors with Array Processors we see that Vector Processors are more powerful computing machines because of utilization of multiple pipe lines on independent stream of data.

4.3. Multi Processors

These are MIMD machines running on small scale. Multiprocessor systems are composed of a number of independent processors. Since these machines are MIMD machines the Multiprocessor Systems have n “Control Units” and n “Processing Units”. Processors are coordinating and co-operating Processors. Multiprocessors are classified into Shared Memory Architecture and Distributed Memory Architecture. Processors cooperate with each other through sharing of data required. It is obvious that multiprocessor based systems are more powerful from point of view of computing power as compared to the single processor based systems. .

4.4. Data Flow Machine

In Data Flow architecture, the instruction execution depends upon the availability of data. That is if data is not available in the machines, instruction will not be executed and if data is available in the machine then instruction will be executed on the available data. The Program Counter Register has the purpose of keeping in record the next instruction executed. Since the instruction execution depends upon the availability of data there is no need of Program Counter Register and Control Flow mechanism in Data Flow machines.

We can mathematically model the Data Flow Machine as follows:

$$I = f(D)$$

Here I represent the Instructions executed on Data Flow Machines and D represents the available Data for the required processing on Data Flow Machines..

4.5. Mas Par Computer (MP-1)

The MasPar MP-1 computer system is massively parallel SIMD machine that is Single Instruction stream Multiple Data stream computer system. It was designed by Digital Equipment Corporation. This is a SIMD array of 4096 processors configured as a square 64×64 array with toroidal wraparound in both directions [1]. Various arithmetic and logical operations can be implemented by this computer system. Integer, rational, real and complex arithmetic operations are implemented by it .There are two type of number representations in MasPar MP-1 computer system: conventional and novel. Its arithmetic system provides binary integer and floating-point computational framework. Some special systems such as Residue Number System RNS is designed in it. Some other type of operations such as logarithmic, level-index and symmetric level index arithmetic operations are in the experimental design phase.

The architecture of MasPar MP-1 Computer System is as follows: The individual processors MasPar system are 4 bit processors. Its all arithmetic is implemented in software. The typical feature of the MasPar system is that at a particular time instant either all processors perform the same instructions or are inactive. Let us consider an example of Matrix Addition on MasPar system. A single instruction on MasPar system can perform addition of two 64×64 matrices. Matrix multiplication is well suited to the array. When we compare the MasPar system with conventional computers we see that it has speed advantage for operations like matrix multiplication over the conventional computer systems. This speed advantage comes from the massive parallelism overcoming the slower individual operations [1].

The MasPar system has marvelous computational power. This 64×64 array of 4-bit processors can be used to simulate hardware implementations of the various arithmetic schemes and to make changes easily in the algorithms being used. It is also possible that we implement the arithmetic using serial algorithm so that the main computation is then spread across the processors. In this way the time penalty inherent in performing such experimental computation can be reduced by taking advantage of the parallelism [4]. For comparing the performance of different arithmetic systems on particular problems, we can implement the standard floating point and integer arithmetic in the similar manner. The significant feature of such arithmetic operations is that built-in arithmetic on MasPar system is “nibble by nibble”. Therefore timing comparisons can be made with some justification. A nibble is defined as half-byte or 4 bits. Since a nibble corresponds to a hexadecimal digit, therefore it is justified to use radix 16 to implement the internal arithmetic of any system.

When we compare the MasPar system’s parallel array with the pipelined vector processors , we see that the MasPar system’s parallel array allows realistic experimental computation without the enormous time-penalties which would be suffered on pipelined vector processors or on conventional serial computers.

5. Systolic versus Wave front Array Processors

Both Systolic and Wave front-array parallel computers are classified as SIMD parallel computers [3]. In both systolic and Wave front processors, each processing element executes the same and only instruction, but on different data.

5.1. Systolic Processors

Systolic Processors consist of a large number of uniform processors connected in an array topology. Each processor usually performs only one specialized operation and has only enough local memory to perform its designated operation, and to store the inputs and outputs [3]. The individual processors or processing elements PE, take inputs from the top and left, perform a specified operation, and output the results to the right and bottom. The processors are connected to the four nearest neighbouring processors in the nearest-neighbour topology.

Processing or firing at each of the cells occur simultaneously in synchronization with a central clock. The name comes from the fact that each cell fires on this heartbeat. Inputs to the system are from memory stores or input devices at the boundary cells at the left and top. Outputs to the memory or output devices are obtained from boundary cells at the right and bottom. Systolic processors are fast and can be implemented in VLSI. They are somewhat troublesome, however in dealing with propagation delays in the connection buses and in the availability of inputs when the clock ticks.

5.2. Wave front Array Processors

Wave front processors consist of an array of identical processors, each with its own local memory and connected in a nearest-neighbour topology. Each processor usually performs only one specialized operation. Hybrids containing two or more different type cells are possible. The cells fire asynchronously when all required inputs from the left and top are present [3]. Outputs then appear to the right and below. Unlike the Systolic processors, the outputs are the unaltered inputs. That is, the top input is transmitted unaltered to the bottom output bus, and the left input is transmitted unaltered to the right output bus. Also, different from the Systolic processor, outputs from the wave front processor are read directly from the local memory of selected cells and output obtained from boundary cells. Inputs are still placed on the top and left input buses of boundary cells. The fact that inputs propagate through the array unaltered like a wave gives this architecture its name.

Wave front Processors combine the best of systolic architectures with dataflow architectures. That is, they support an asynchronous data flow computing structure; timing in the interconnection buses and at input and output devices is not a problem. Furthermore, the structure can be implemented in VLSI.

5.3. Tabular Comparison of Systolic and Wave front Array Processors

Following is the detailed comparison of Systolic Processors and Wave front Array Processors. We highlight the similarities and differences of these processors specific to qualitative parameters.

Table1. Similarities of Systolic and Wave front Array Processors

PARAMETERS	SYSTOLIC	WAVEFRONT
Processor	Uniform	Identical
Topology	Array	Array
Operation	Only 1 specialized operation	Only 1 specialized operation
Memory	Local memory	Local memory
Inputs	From top and left	From top and left
Processor Type	Same	Hybrid processors are possible containing different type of cells
Outputs	Processor performs specified operation and output result to right and bottom	Outputs are unaltered inputs
Input Processing	Inputs are accepted from top and left and then processed by processor or processing element	Top input is transmitted unaltered to the bottom output bus and left input is transmitted unaltered to the right output bus

Table 2. Differences of Systolic and Wave front Array Processors

PARAMETERS	SYSTOLIC	WAVE FRONT
Processor Type	same	Hybrid processors are possible containing different type of cells
Outputs	Processor performs specified operation and output result to right and bottom	Outputs are unaltered inputs
Input Processing	Inputs are accepted from top and left and then processed by processor or processing element	Top input is transmitted unaltered to the bottom output bus and left input is transmitted unaltered to the right output bus
Clock	Synchronized with clock	Not synchronized with clock
Input Source	Inputs to the system are from memory stores or input devices at the boundary cells at the left and top	Inputs to the system are still placed on the top and left input buses of boundary cells
Data Flow	Synchronous data flow	Asynchronous data flow

.

6. CONCLUSIONS

The above survey of different parallel computing machines leads us to conclude that although we have diverse range of Parallel Processors and every Parallel Processing based machine has its own significance but it depends upon the type of problem to be solved by parallel processing based computer system. In this research paper we have elaborated the concept of Pipelining since the in depth understanding of Pipelining is indispensable to compare different paradigms of parallel processing based computer systems. Then we have described Pipelining specifically in parallel processing based computer systems. Then we compared the Reduced Instruction Set Computers (RISC) with Complex Instruction Set Computers (CISC). After detailed comparison of different parallel computing machines we see that Array Processors are parallel computer systems having single control unit. Array Processors are subclass of SIMD computers. Vector Processors perform computation through Vectors. The Vector Processors have more computing power as compared to Array Processors and other parallel processing based computer system because of employing multiple pipelining which gives strength to computing power of Vector

Processors. Multiprocessors are MIMD machines in which processors are coordinating and cooperative processors because processors cooperate with each other through sharing of data required. The Optimization of Register use is the typical hall mark of a RISC machine. Data Flow machines work on the availability of data and there is no significance and need of Program Counter register and control flow mechanism in Data Flow machines. Then we have described the Mas Par MP-1 system in detail. The Mas Par MP-1 system is SIMD based machine employing an array of 4096 processors. Each processor is of 4 bit size. The Mas Par system has exceptional computing power. If computational complexity of a problem is increased then the MasPar MP-1 systems are most suitable for solving such problems. Because of employing 64 × 64 array of processors this system is suitable for performing matrix operations. Since we have a range of fields in applied sciences and engineering which contain scientific problems involving mathematical operations on matrices of very large order and size. Such matrices cannot be solved on serial conventional computer systems. The MasPar MP-1 systems should be utilized for solving problems involving matrices of very large size in different fields of science and engineering. Also a wide range of computer arithmetic operations including integer, real, rational and complex arithmetic is implemented by the software arithmetic laboratory of the DEC/MasPar MP-1 system .The Mas Par systems are specially designed to simulate hardware implementations of the various arithmetic schemes and to make alterations easily in the algorithms being used. Also we have compared two SIMD machines Systolic Processors and Wave front Processors. We have highlighted the similarities and differences between Systolic Processors and Wave front Processors. When we compare Systolic Processors with Wave front Processors we see that regarding the speed of operation, Wave front Processors are better than Systolic Processors.

ACKNOWLEDGEMENTS

We are very much grateful for the work on Pipelining by Gilreath W. and Laplante [5]. Multiprocessor based systems and Multi computer systems have been evolved in the recent years. Today the i- core multiprocessor based systems are the most advanced forms of SIMD/MIMD systems. Indeed the Parallel Processing has given scintillating growth and advancement to the field of faster computer designs. We are thankful to the work of Gilreath W. and Laplante P.[5] for the work on Systolic Processors and Wave front Processors. Also John P. Hayes gives a very comprehensive overview of Systolic Processors and demonstrated matrix multiplication algorithm on Systolic Arrays [6].William Stalling [2] also gives the comprehensive overview of Pipelining, stages of Pipelining and Pipelining in RISC and CISC machines. John P. Hayes [6] also gives a detailed description of Pipelining, RISC machines and CISC machines. A huge contribution to the work on Computer Arithmetic Laboratory Software of the DEC /MasPar MP-1 system is done by Michael A. Anuta (Cray Research Inc., Calverton, MD 20705) , Daniel W. Lozier (National Institute of Standards and Technology, Gaithersburg, MD 20899-0001) and Peter R. Turner (United States Naval Academy, Annapolis, MD 21402) [1].

REFERENCES

[1] Michael A. Anuta. Daniel W. Lozier, and Peter R. Turner, "The MasPar MP-1 As a Computer Arithmetic Laboratory," Journal of Research of the National Institute of Standards and Technology, vol. 101, no. 2, pp. 165-174, March –April 1996..
[2] William Stalling, Computer Architecture and Organization, 3rd ed., McGraw Hill: UOS Press, 2004, pp.6-9.
[3] Philip A. Laplante, Real Time Systems, 3rd ed., Wiley Inter-Science, IEEE press: September 2003, pp. 66 -68
[4] Kai Hawang , Advanced Computer Architecture and Organization: Parallelism, Scalability, Programmability, 1st ed., McGraw Hill Science/Engineering/Math: ,December 1 1992, pp.6-9.
[5] Gilreath W., Laplante, P., Computer Architecture: A Minimalist Perspective, Kluwer Academic Publishers, Norwell MA 2003
[6] John P. Hayes, Computer Architecture and Organization, 3rd ed., McGraw Hill Publisher, 1997
[7] Albert Y. H. Zomaya, Parallel and Distributed Computing Handbook, 3rd ed., Mc Graw Hill Series on Computer Engineering, New York NY ,USA,1996 , pp. 66 -68
[8] Jerry Fiddler, Eric Stromberg , David N. Wilner, "Software considerations for Real Time RISC," in Compcon Spring '90 ,Intellectual Leverage, Digest of Papers, Thirty-Fifth IEEE Computer Society International Conference. Page(s): 274 – 277, 1990

AUTHORS

Ahmed Faraz has done Bachelor of Engineering degree in Computer Systems and Masters of Engineering in Computer Systems from NED University of Engineering and Technology Karachi in 1998 and 2004 respectively. He served Sir Syed University of Engineering and Technology as Lecturer for six years from 1998-2004.He served NED University of Engineering and Technology Karachi as Assistant Professor for 2 years. Currently He is currently working as Assistant Professor at Department of Computer and Software Engineering of Bahria University Karachi Campus. His research interest include Parallel Processing, Distributed computing, Artificial Intelligence, Digital Signal Processing, Image Processing and Computer Vision

Faiz Ul Haque Zia has done Bachelor of Engineering degree in Computer Systems Engineering from NED University of Engineering and Technology Karachi in 1998. He also got Masters of Science degree in Computer Science from University of Tulsa in 2003. He has vast teaching experience at Bahria University Karachi Campus. He is currently working as Associate Professor at Department of Computer and Software Engineering of Bahria University. His research interests include Artificial Intelligence, Data Mining, Natural Language Processing and algorithms.

Majid Kaleem has done Bachelor of Computer Engineering from Adamson University Philippines in 1995. He also got Masters from University of Sunderland UK in 2007. He has vast teaching and administrative experience. He is currently working as Assistant Professor and Head of the Department of Computer and software Engineering Bahria University Karachi Campus.