Analysis of Parallel Architectures: SIMD, tightly-coupled MIMD, and loosely-coupled MIMD

Babasegun Adeleye[#1], Salman Mohammed Jiddah[#2]
Department of Computer engineering, Cyprus International University
Lefkosa, Mersin 10, Turkey

Abstract- *This paper looks into the two forms of parallel computing which are Single Instruction Multiple Data (SIMD) and Multiple Instruction Multiple Data (MIMD), and further diving into the sub categories of MIMD; loosely coupled and tightly coupled with an overview of their individual architectures and their implementation in the computing industry. For a case study of their industrial application, this paper looked into the Intel Iris processor as a practical implementation of SIMD architecture in the industry, IBM Power8 for tightly coupled MIMD systems which is a supercomputer, and lastly the Beowulf cluster system which is an implementation of loosely coupled MIMD. Each architecture implementation is looked into regarding why it is being chosen to be used in the industry.*

I. INTRODUCTION

The need for increase in performance of computer systems cannot be overestimated, as it enhances instruction processing. Parallel processing works by dividing large problems into smaller parts which are then solved simultaneously. Ideally, it makes instruction execution faster because there are multiple processors working on the program, but it's often difficult to divide an instruction in a way that separate CPUs can execute different portions without interfering with each other. As a result, it has become a dominant paradigm in computer architecture that devices have multi-core processors. With multi-core processors, instruction execution can be done simultaneously by each processor. These processors handle instruction execution differently based on their architecture.

Michael J. Flynn in his Flynn taxonomy classified parallel computer architectures into four based on the number of instructions and data they can handle at time [1]. The main focus here is on two architectures which are as follows:

- Single Instruction Multiple Data(SIMD)
- Multiple Instruction Multiple Data(MIMD)

II. Single Instruction Multiple Data

Single Instruction Multiple Data (SIMD) is a parallel architecture which carries out a single action on multiple sets of data. Here, multiple processors work on several data items from a particular instruction set simultaneously using various processing elements. Machines with SIMD architecture are targeted toward applications that implement data level parallelism.

The processing units of SIMD processors arean array of fined-grained that is connected in a network topology. The processor array is connected to a control protocol responsible for fetching and interpreting instructions.

Examples of architectures that implements SIMD architecture are Graphics Processing Units GPU, Database searches and genetic sequence matching. The major architecture of focus here is the Intel Iris graphics processor which is a type of Graphics Processing Unit.

A. *Intel Iris Graphics*

Intel Iris Graphics processing unit is created by Intel Corporation. It is mostly used in new versions of Intel processors like the Intel core i5 and on some low power form factor like Intel Core M processor [14].

Intel Iris execution unit combines simultaneous multithreading with fine-grained interleaved multithreading [14].These components support multiple SIMD and ALU pipelined across multiple threads which increases throughput when computing floating points and integer. Due to the threaded nature of the execution unit, instructions are always available to execute, also the latency of operations which take longer to execute is hidden [14]. Depending on the software workload, the threads within the Execution Unit are either executing codes from different compute kernel or from the same compute kernel.Every cycle, the Execution Unit thread can issue as much as four different instructions; these instructions are selected from four different threads.

In each EU, a pair of SIMD floating points is the major components used for computation. The SIMD units can carry out executions ranging from four 32-bit operations to eight 16-bit operations [14].

III. Multiple Instructions Multiple Data

Multiple Instructions Multiple Data is a type of parallel architecture which its most basic goal is to achieve parallel computing and also the most known type of parallel processing system. It may consist of independent or tightly coupled processors each with memory that can be accessible to all processors or a local and not directly accessible by other processors.

There are two types of MIMD processors, namely:

- Tightly Coupled MIMD
- Loosely Coupled MIMD

A. Tightly Coupled MIMD

These are shared memory multiple instruction multiple data systems. Processors also share I/O devices and are connected by a bus. The entire system is controlled by a single Operating System which provides interaction between processors and their programs at job, file and data task levels. This interconnection makes it easy for communication among processors,and helps speed up instruction execution. in the category of tightly coupled MIMD there are two sub categories which are Symmetric Multiprocessor(SMP) and Non-Uniform Memory Access (NUMA).

1) Symmetric Multiprocessor

Symmetric multiprocessing is a kind of programs processing using multiple processors sharing the same operating system and memory, as well as input and output bus. An example of a system implementing this architect is the IBM POWER8.

2) IBM POWER8

IBM Power8 has the record of being the first-generation big data cloud hardware to be designed, its design was prompted by the enormous amount of data being generated daily across the globe, the IBM Power 8 was designed as an enormous chip which is multithreaded with a 12-core chip, each of the cores is able to simultaneously handle a total of eight threads.

3) Non-Uniform Memory Access

This is a type of tightly coupled MIMD architecture which is designed to handle multiprocessing based on the relativity between of a process and a memory, as a process accesses its local memory faster than a non-local memory which is a memory shared amongst processors. An example of a system implementing this architecture is the Intel Itanium.

I) Intel Itanium

Intel Itanium are microprocessors that implement the IA-64 architecture which are primarily marketed to high end computing such as enterprise servers, the architecture of the Intel Itanium was designed by fusingRISC and VLIW best features, which makesit a hybrid of the two. Having a floating-point architecture, with its floating-point data types having an accuracy of 64-bit with a 17-bit exponent making it internally mapped to 82-bit format.

B. Loosely Coupled MIMD

These are essentially distributed memory multiple instructions multiple data systems. Every processor in the system has its own individual memory, and all processors have no direct access to the other processors memory in the system. Loosely coupled systems have advantages over tightly coupled systems such as the automatic scaling up of bandwidth of the system with respect to the number of processors which can in turn be used to solve the speed advantage tightly coupled systems have over it by scaling higher. The fore mentioned advantage still leaves the loosely coupled MIMD with its biggest disadvantage of being slower than tightly coupled systems because of the slow communication between processors due to having different memory independently. A system which models this architecture is the Beowulf Cluster.

1) Beowulf Cluster

Beowulf Cluster is an architecture that is composed of multiple computers connected and can be used for parallel computing. This architecture is composed of a server node and a single or multiple client nodes which are connected through an Ethernet or over a network. These nodes are clustered set of computers dedicated to running high performance computing task. Mostly built on UNIX operating systems, there is a client server relationship amongst the computers connected. The server node which usually controls the whole cluster serves files and data to other client nodes in cluster and acts as the main access of the cluster to the outside world.

Applications like Parallel Virtual Machine (PVM) and Message Passing Interface (MPI) are used in the cluster. The PVM is a tool which designed to allow a network of heterogeneous UNIX or Windows machines used as a single distributed parallel processor. Communication among nodes in the cluster is usually done through the use of a Message Passing Interface (MPI) because the nodes can’t communicate directly or access each other’s memory. This MPI is an application which runs on the cluster and works by sending requests to other nodes to confirm if they have the message needed by the source node. If any node has the data required, the data is then transferred to the node that needs the data.

IV. Conclusion

Each architecture is designed to process information different ways. Depending on what is to be achieved, the choice to select a specific architecture largely depends on what kind of computing is to be done, and what resources are available.

References

[1] Flynn, Michael J. "Very high-speed computing systems." Proceedings of the IEEE 54.12 (1966): 1901-1909.

[2] Kaur, Mandeep, and RajdeepKaur. "A comparative analysis of SIMD and MIMD architectures." *International Journal of Advanced Research in Computer Science and Software Engineering* 3.9 (2013).

[3] Adve, Sarita V., and KouroshGharachorloo. "Shared memory consistency models: A tutorial." *computer* 29.12 (1996): 66-76.

[4] Hord, R. Michael. *Parallel supercomputing in MIMD architectures*. CRC press, 1993.

[5] Sampath, S., Bharat BhushanSagar, and B. R. Nanjesh. "Performance evaluation and comparison of MPI and PVM using a cluster based parallel computing architecture." *Circuits, Power and Computing Technologies (ICCPCT), 2013 International Conference on*. IEEE, 2013.

[6] Luo, Wen-lang, An-dong Xie, and Wen Ruan. "The construction and test for a small beowulf parallel computing system." *Intelligent Information Technology and Security Informatics (IITSI), 2010 Third International Symposium on*. IEEE, 2010.

[7] Zhu, Yongzhi, and Baoxiang Cao. "A Scalability Metric Based on Beowulf Cluster System." *Distributed Computing and Applications to Business Engineering and Science (DCABES), 2010 Ninth International Symposium on*. IEEE, 2010.

[8] Becker, Donald J., et al. "BEOWULF: A parallel workstation for scientific computation." *Proceedings, International Conference on Parallel Processing*. Vol. 95. 1995.

[9] Bhuyan, Laxmi N. "On the performance of loosely coupled multiprocessors." *ACM SIGARCH Computer Architecture News* 12.3 (1984): 256-262.

[10] Liu, Yingying, Dake Liu, and Wei Wang. "IIR parallelization on multi-datapath SIMD architecture." *Integrated Circuits and Microsystems (ICICM), International Conference on*. IEEE, 2016.

[11] Sharangpani, Harsh. "Intel® Itanium™ processor microarchitecture overview." *Microprocessor Forum*. 1999.

[12] Lameter, Christoph. "Numa (non-uniform memory access): An overview." *Queue* 11.7 (2013): 40.

[13] Sinharoy, Balaram, et al. "IBM POWER8 processor core microarchitecture." *IBM Journal of Research and Development* 59.1 (2015): 2-1.

[14] Junkins, Stephen. "The Compute Architecture of Intel® Processor Graphics Gen9." *Intel whitepaper v1* (2015).