TECHNICAL REPORT

COMP-2660

COMPUTER ARCHITECTURE II: MICROPROCESSOR PROGRAMMING

Parallel Computer Architecture Design

Author:
Rocio Rueda (110102315)

Submitted to:
Reem Al-Saidi

October 17, 2023

"I confirm that I will keep the content of this assignment confidential. I confirm that I have not received any unauthorized assistance in preparing for or writing this assignment. I acknowledge that a mark of 0 may be assigned for copied work."

Parallel Computer Architecture Design

1st Rocio Rueda
School of Computer Science
University of Windsor
Windsor, Canada
ruedar@uwindsor.ca

I. Introduction

Technologies over time are getting more complex, powerful, and efficient in its use. With technologies improving dramatically over time, this presents more challenges. A new revolution in technology started in the 1950s with computers like the Universal Automatic Computer (UNIVAC), which was one of the earliest commercial computers ever made. This computer was designed as a commercial data-processing computer that could handle both alphabetic and numeric data [24]. In 1994, IBM released the ThinkPad 775CD, which was the first laptop that featured an integrated compact disc read-only memory (CD-ROM) [25]. Current laptops focus on improving processing speeds, thinner design and better refresh-rates.

Parallel computers add a new dimension in the development of a computer system by using a greater number of processors [16]. The cores are integrated into a "single chip package or onto a single integrated circle die, and several architectures such as multithreading, superscalar, vector, or Very long instruction word (VLIW)" [22], which allow to perform more than one basic instruction at a time [23].

In this regard, this article explains the importance of parallel computer architecture systems. The outcomes of parallel systems design used to aid higher computational performance are also discussed [1]. The article also goes into deeper details on how parallel architectural design helps with software applications like data distribution. This article also discusses the different kinds of parallel architectures that exist and the relation among them, including their improvements, advantages and disadvantages. Lastly, the different types of computers that experiment or use specific parallel computer architecture designs are discussed, as well as a comparison with recent parallel computer architectures.

II. Importance

This topic is relevant to the field of computer science and the assembly programming language because due to the continuing changes of computer architectures over the years, computer architectures from cellphones to supercomputers will need to use parallelism to some extent [1]. Parallel computer architectures aids with the full potential of using manycore technology. Manycore processors is an integrated circuit that contains multiple processor cores, which results in a higher degree of parallel processing. This, in turn, makes the computer more efficient because it allows more work to be done, as since multiple tasks are being done at the same time, which results into shorter execution time.

Identify applicable funding agency here. If none, delete this.

A. Software Perspective

On the software side, core mechanisms for this architecture are distributed data management, distributed data sharing, and distributed learning [3]. This is because parallel computer architectures involve multiple nodes or processors that work together to produce large datasets. Blockchain is important in parallel computing architecture to aid in building these larger data sets [3]. The authors of [3] stated that they would like to use large data set from various hosted medical data sets that would in turn, enable research to jump start deep learning research for medical domains [3]. Blockchain architectures consist of designing a structure as a peer-to-peer (P2P) network, which is served in backend systems and applications [4]. The four primary types of blockchain architectures are public, private, consortium and hybrid. Blockchain was originally designed as a duplicated computing architecture that every blockchain node running identical code with the same ledger [3]. Transforming this duplicated computing into distributed parallel computing relies on smart contracts, which are a collection of methods that each act upon a particular set of state variables [17]. This article stated due to the "lack of distributed and parallel computing deep learning mechanisms for heterogeneous and distributed data", their goal was to use smart contract to "achieve the patient centric medicine by exploring standardized, trusted and secure data exchange methodologies with deep learning analysis for personal healthcare and precision medicine" [3].

B. Hardware Perspective

From the hardware perspective, parallel computer architectures aid in the changes of the computer system, especially since we are currently moving to smaller chips, which increase the likelihood of defects to occur. The following remedies are suggested according to the "Berkeley View" report:

Power: Parallelism leads to energy-efficient processors. The main reason for this is that parallelism reduces the clock frequency and power supply voltage of a system [10]. In addition, "many simple cores offer higher performance per unit

area for parallel codes than a comparable design employing smaller numbers of complex cores" [1].

Design cost: Simpler processing elements are easier to predict their behaviour than complex processing elements, due to their simpler design. In addition, they are more compliant to formal verification [1]. The lower complexity a chip results in being more economical to be designed and produced.

Defect tolerance: The smaller the processing elements are, the more economic it is to improve defect tolerance. This is because parallel computer architectures generally provide many redundant cores, and hence if there is a defect, they can be turned off. An example of well-known branded chips with redundant cores is the Cisco Metro chip that has four redundant processor cores per die. Another example is the STI Cell processor, which has eight cores. However, six are enabled in the Sony PlayStation 3 mainstream consumer application to provide additional redundancy for tolerating more defects [1].

In addition to the benefits listed above, companies like Intel and AMD use roadmaps that indicate tighter integration between GPUs and CPUs, which is the most likely path toward introducing manycore processing to mainstream consumer applications on desktop and laptop computers [1].

III. Technological Details

According to *The new landscape of parallel computer architecture* written by John Shalf, it explains "how current constraints on device physics at the silicon level would affect CPU design, system architecture, and programming models for future systems" [1]. The graph depicted in Figure 1 shows Moore's law, which states "doubling the density of components per integrated circuit at regular interval" [14]. This graph is used to show that ILP and clock frequencies have been flattening.

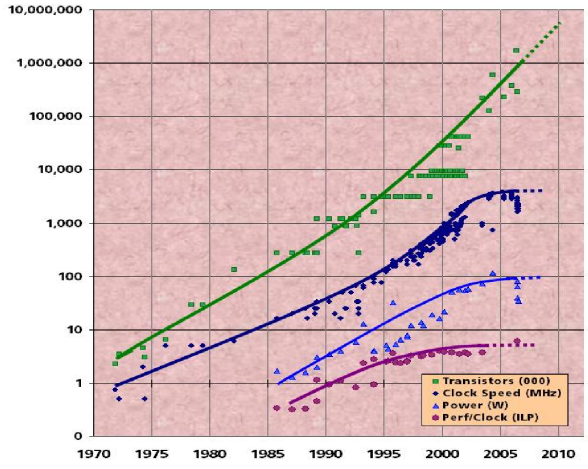

Fig. 1. Graph from Kunle Olukotun and Herb Sutter demonstrate Moore's Law shows demonstration of Moore's Law and traditional sources of performance improvements have been flattening [1]

A. Manycore and Multicore

The article mentioned in the previous paragraph goes more in detail with their plan to use geometric growth in system concurrency of interconnect design, memory balance, and I/O system design that will change the High-Performance Computing design (HPC) to Massively Parallel Processors (MPPs) [1]. Two important CPUs are mentioned, creating a new "generation of parallel computers". These are the multicore and manycore processors. It began with multicore, which the goal was "doubling the number of standard cores per die with every semiconductor process generation starting from a single processor" [1]. Multicore processors are also good for their potential in processing capabilities, and their hardware performance [18]. However, in order to achieve high parallel workloads, the trajectory changed to manycore which was created with simpler cores and lower clock frequencies [1].

B. Shift to Newer Technologies

In the early 2000s, handheld consumer electronics became more popular in the market. This trend created significant implications for the High-Performance Computing design (HPC). Below, Figure 2 shows the "market share of consumer electronics applications for CPUs surpassed that of the desktop PC industry in 2003" [1].

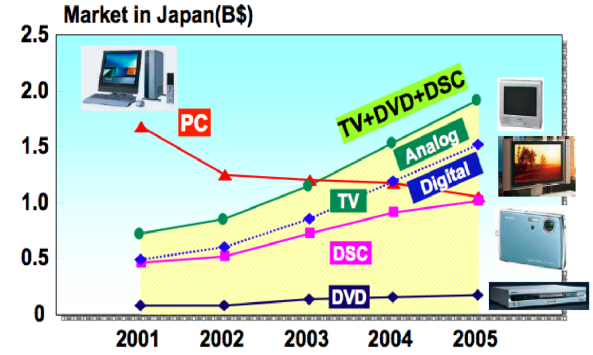

Fig. 2. Declining influence of PC microprocessors on the overall revenue share in microprocessor-based electronic designs (from Tsugio Makimoto: JEITA) [1]

The charts depicted in the figure represents the declining influence of PC microprocessors on the overall revenue per year of each electronic device. The market share of the consumer electronic applications for CPUs surpassed that of the desktop PC industry in 2003. Portability increased its popularity as Apple released the iPod, a device used for playing music, and then soon after cellular phone devices. This led on constructing new chip designs from embedded applications rather than the transistor. Overtime, CPU cores become simpler with HPC applications, slowly becoming a replacement for transistors that form the CPU. The article predicted that, "within the next 3 years, there will be manycore chip designs that contain greater than 2000 CPU cores, which is very close to the number of transistors that we used in

the very first Intel 4004 CPU" [1]. This trend is in line with what Chris Rowen said of the new electronic design prediction, stating that the "processor is the new transistor" [1].

C. OpenMP and MPI Programming Models

On the software side, the OpenMP programming model come in impact circa 1996 [19], and MPI around 1992 [20]. Since multicore and manycore Cellular Multi-Processing (CMP) is an unfamiliar programming target, the most popular approach was to program familiar targets such as Massively Parallel Processing (MPPs) or Symmetric Multiprocessing (SMPs). MPI programming model was used for MPPs. However, the implementations grew to either O(N) or O(N^2) with the number of processors consuming too much memory [1]. The OpenMP and MPI programming models have shown more failures than successes, particularly, in the deficiencies of the implementation. In addition, SMPs and OpenMPs have dramatically different characteristics than CMPs. In addition, CMP offers 10-100 times lower latencies and 10-100 higher bandwidths between the CPU cores on the chip. This is one of the reasons why multicore and manycore processors become essential components in modern CPUs and GPUs.

D. The Building of Parallel Architectures

In the past most architectures had only the following features:

- Instruction pipelining: This decomposes of instruction execution into autonomous stages in a linear fashion [3], which performs only a process of execution process.
- Multiple CPU functional units: this provides independent functional units for arithmetic and Boolean operations that are executed concurrently [3].
- Separate CPU and I/O processors: frees the CPU from the responsibilities of I/O using allocated I/O processors [3].

*1) **Synchronous Parallel Architectures - Pipeline Vector Processors**:* Synchronous parallel architectures started being developed as early as the late 1960s and early 1970s, where pipelined vector processors were developed [5]. These were multiple, pipelined functional units that provided parallel vector processing by sequentially streaming vector elements to perform a process called "chaining". Chaining is a "dependence analysis where the links represent the dependence relationships that exist in architectural specification" [6]. Supercomputers such as the Cray X-MP/4 and ETA-10 unite four to 10 vector processors through a large-scale memory. Figure 3 shows vector pipeline with addition operation:

*2) **Synchronous Parallel Architectures - SIMD Architectures**:* Secondly, SIMD architectures which employ a CPU, multiple processors, and an interconnection network for either processor-to-process or processor-to-memory communications [5]. SIMD execution is structured in processor arrays that have been employed for large-scale scientific calculations,

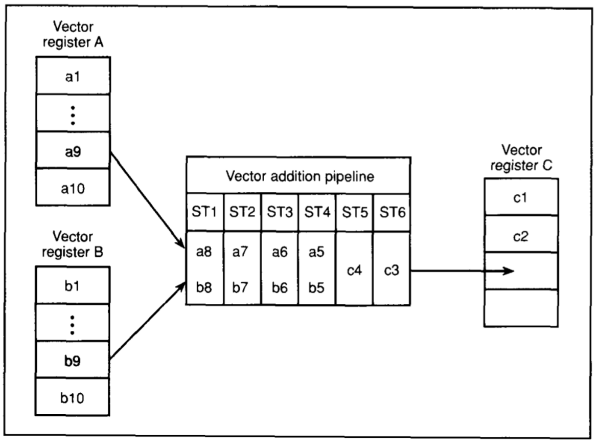

Fig. 3. Register-to-register vector architecture operation [6]

an example being image processing. Loral's Massively Parallel Processor and ICL's Distributed Array Processor exemplify image processing applications by mapping pixels to the memory's planar structure [5]. The SIMD architecture uses associative memory processor to access stored data in parallel. This processor reads and executes instructions, using an array controller when associative memory instructions are encountered. Figure 4 shows the characteristic functional units of an associative memory processor.

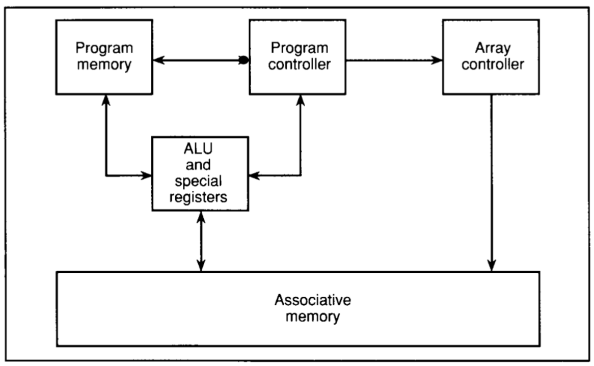

Fig. 4. Associative memory processing organization [6]

*3) **Synchronous Parallel Architectures - Systolic Architectures**:* Thirdly, systolic architectures solve problems of special-purpose systems that balance insensitive computations with demanding I/O bandwidth. These architectures are pipelined multiprocessors whose data is pulsed from memory and through a network of processors. The systolic arrays avoid I/O and memory bandwidth bottlenecks to achieve significant parallel computations. They also achieve this by obtaining pipeline data through multiple processors. Systolic arrays need rhythmic data flow, and zeros in random positions, which cannot be removed for making utilization better [11].

*4) **Asynchronous Parallel Architectures - MIMD Architectures**:* An example of an asynchronous parallel architecture

is the MIMD architecture. These architectures can execute independent instruction streams. They use multiple instructions that are built on multiple data to boost the performance of the computer, and work with programming models such as shared memory and distributed memory. These architectures are one of the most recent and popular computer architectures used in parallel computers.

E. Parallel Computers

1) ***Lawrence Snyder's Configurable Highly Parallel Computer****:* The Lawrence Snyder's Configurable Highly Parallel Computer, also known as CHiP, uses reconfigurable topology architecture to support architectural expandability in distributed memory architectures, which MIMD architecture uses. This computer provides programmable interconnection structure integrated with the processing elements. It is constructed from a collection of homogenous microprocessors, a switch lattice, and a controller. The switch lattice is defined as a regular structure formed from programmable switches connected by data paths [9]. Figure 5 shows the three examples of switch lattices.

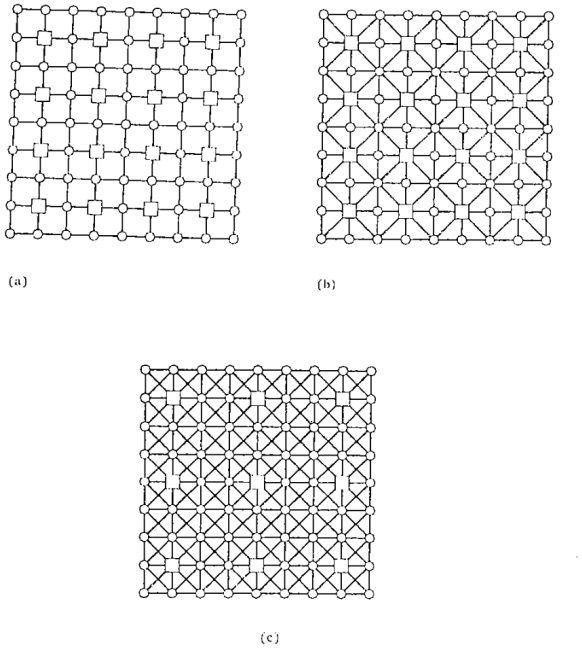

Fig. 5. Three switch lattice structures. Circles represent switches; squares represent Processing Elements (PE) [6]

2) ***AP1000****:* Another parallel computer, which uses distributed memory, is known as the AP1000. It ensures that each processor runs its own program from local memory and communicates with other processors by passing massages. The goal for this computer is to reduce overall communication latency [7]. It is composed of 1,024 processors and uses the MIMD architecture [13]. Figure 6 shows the network structure of AP1000 computer.

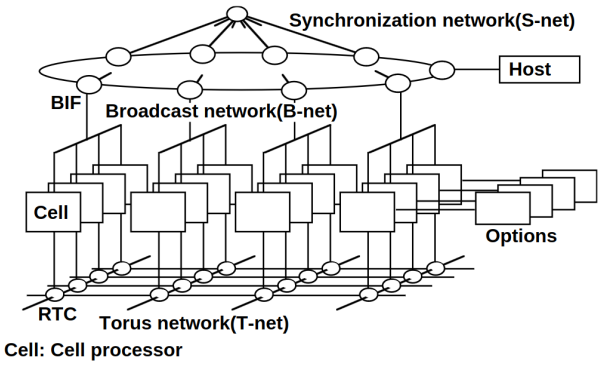

Fig. 6. AP1000 system configuration [7]

3) ***EM-X****:* Lastly, the EM-X parallel computer uses communication and synchronization mechanization methods, which reduce latency, and use multi-threading, which supports share memory, an important model in MIMD architectures [8]. The architecture uses very simple packets for inner processor communications, where it uses packet prioritization to support flexible packet scheduling. Figure 7 shows the paths of the packets in a processing element.

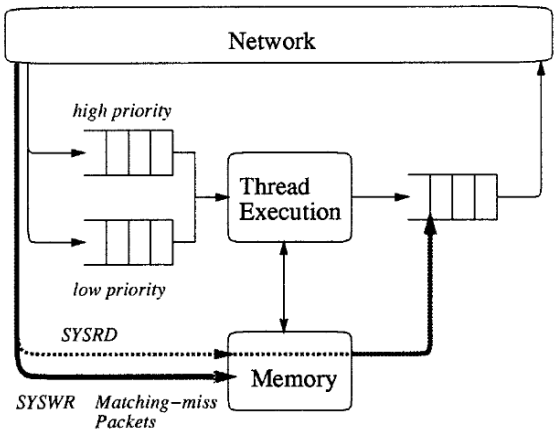

Fig. 7. Flow Packets of Processing Elements [8]

F. An Overview

Although these computers mentioned were design 20-30 years ago, they exemplify characteristics of parallel computer architecture systems today. Parallel computer architecture systems today use a wide range of systems, such as multi-core, manycore, and/or vector processors, cluster computing, distributive and quantum computing.

IV. Discussion

There are seemingly a lot of benefits regarding parallel computer architectures, one being to stride away from the efficient silicon implementation of traditional uniprocessor architectures, however, there are some disadvantages. Firstly,

with the manycore processors that are designed for aiding in parallel processing. The manycore design packs so many cores onto a chip that destabilize system balance, mostly with memory bandwidth [1]. This can lead to being more prone to failures in the system. The good news is that there are methods mentioned in this article to combat this problem, one being the defect tolerance method which contains multiple redundant cores in the processor such that they can be turned off if there is a deficiency [1]. Another disadvantage revolves around the MIMD architecture. MIMD architectures are currently the most popular and efficient architectures for parallel architecture design, since there is no need to partition either the code or data [15]. There are some disadvantages though. MIMD architectures require more memory than the corresponding SIMD [11], scalability beyond thirty-two processors is difficult, and the shared memory model is less flexible than the distributed memory model [12]. Overall, developers are always finding ways to make computer systems more efficient, cost-effective, and powerful, and parallel architectures will only improve over the years.

V. Conclusion

In conclusion, parallel computer architectures are still an important topic today as it is constantly changing. The parallel architecture design can be beneficial in both hardware and software, as well as for data management and machine learning applications, since it allows computers to execute code more efficiently. Since electronics are changing rapidly over the decades, improving current computer architecture is crucial to better enhance performance in future technologies. In addition, different parallel architectures have better uses and vice versa. For example, if one is looking for more parallelism but less flexibility, SIMD architectures are better. If one is looking for more flexibility and less parallelism, MIMD architectures are better [21]. The same goes for single-core vs multi-core processors. If one does not need the complexity of a multi-core processor and needs to reduce costs, single-core processors are better. Manycore processors came into use for it unleashing the full potential of parallel computer architecture. This is because of it being to employ simpler cores running at modestly lower clock frequencies. The improvements of parallel computer architectures have allowed for more energy-efficient, cost-efficient, and design tolerance system. Different parallel computers that are made nowadays such as the AP1000, whose goal was to reduce communication latency and the EM-X parallel computer which supported shared memory. Overall, parallel computer architectures will allow future hardware to enhance allowing to solve more complex issues in technology.

References

[1] Shalf, J. (2007, July). The new landscape of parallel computer architecture. In Journal of Physics: Conference Series (Vol. 78, No. 1, p. 012066). IOP Publishing. Accessed 16 October 2023.

[2] Takano, S. (2021, April 1). Traditional microarchitectures. Thinking Machines. https://www.sciencedirect.com/science /article/abs/pii/B9780128182796000128. Accessed 16 October 2023.

[3] Shae, Z., & Tsai, J. (2018, July). Transform blockchain into distributed parallel computing architecture for precision medicine. In 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS) (pp. 1290-1299). IEEE. Accessed 16 October 2023.

[4] October, 17. A Guide to Blockchain Architecture. Casper Network. (n.d.). https://casper.network/en-us/web3/blockchain/a-guide-to-blockchain-architecture/. Accessed 16 October 2023.

[5] Duncan, R. (1990). A survey of parallel computer architectures. Computer, 23(2), 5-16. Accessed 16 October 2023.

[6] Stafford, J. A., Richardson, D. J., & Wolf, A. L. (1997). Chaining: A software architecture dependence analysis technique (pp. 1-12). Technical Report CU–CS–845–97, Department of Computer Science, University of Colorado, Boulder, Colorado. Accessed 16 October 2023.

[7] Ishihata, H., Horie, T., Inano, S., Shimizu, T., & Kato, S. (1991, May). An architecture of highly parallel computer AP1000. In IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (Vol. 6). Accessed 16 October 2023.

[8] Kodama, Y., Sakane, H., Sato, M., Yamana, H., Sakai, S., & Yamaguchi, Y. (1995). The EM-X parallel computer: Architecture and basic performance. ACM SIGARCH Computer Architecture News, 23(2), 14-23. Accessed 16 October 2023.

[9] Snyder, L. (1980). Introduction to the Configurable, Highly Parallel (CHiP) Computer. Accessed 16 October 2023.

[10] Wanhammar, L. (1999). Parallelism in Algorithms - DSP Integrated Circuits. ScienceDirect. https://www.sciencedirect.com/topics/computer-science/parallelism. Accessed 16 October 2023.

[11] Das, R., & Krishna, T. (2018, September 17). DNN accelerator architecture – SIMD or systolic?. SIGARCH. https://www.sigarch.org/dnn-accelerator-architecture-simd-or-systolic/. Accessed 16 October 2023.

[12] Introduction to MIMD architecture - university of lucknow. (n.d.). https://www.lkouniv.ac.in/site/writereaddata/siteContent/20200426130637362 0rohit_engg_MIMD_ARCH.pdf. Accessed 16 October 2023.

[13] Kato, S., Murakami, S., Utsumi, Y., & Mizutani, K. (1993). Application of massive parallel computer to Computational Wind Engineering. Computational Wind Engineering 1. https://www.sciencedirect.com/science/article/abs/pii/B9780444816887500437. Accessed 16 October 2023.

[14] Schaller, R. R. (1997). Moore's law: past, present and future. IEEE spectrum, 34(6), 52-59.

[15] Ginni. (2021, July 23). What are shared memory mimd architectures?. Tutorials Point. https://www.tutorialspoint.com/what-are-shared-memory-mimd-architectures. Accessed 16 October 2023.

[16] Parallel Computer Architecture - Quick Guide. Tutorials Point. (n.d.). https://www.tutorialspoint.com/parallel_computer_architecture/parallel_computer_architecture_quick_guide.htm. Accessed 16 October 2023.

[17] Smart contract architecture¶. Smart contract architecture - AElf release/1.2.3 documentation. (n.d.). https://docs.aelf.io/en/latest/architecture/smart-contract/architecture.html. Accessed 16 October 2023.

[18] Bigelow, S. J. (2022, March 7). What is a multi-core processor and how does it work?. Data Center. https://www.techtarget.com/searchdatacenter/definition/multi-core-processor. Accessed 16 October 2023.

[19] de Supinski, Bronis R., Scotland, Thomas R. W., Duran, Alejandro, Klemm, Michael, Bellido, Sergi Mateo, Olivier, Stephen Lecler, Terboven, Christian, & Mattson, Timothy G.. The Ongoing Evolution of OpenMP. United States. https://doi.org/10.1109/JPROC.2018.2853600. Accessed 16 October 2023.

[20] Gillis, A. S., & Bigelow, S. J. (2022, July 29). What is message passing interface (MPI)?. Enterprise Desktop. https://www.techtarget.com/searchenterprisedesktop/definition/message-passing-interface-MPI. Accessed 16 October 2023.

[21] Difference between SIMD and Mimd. Tutorials Point. (n.d.-a). https://www.tutorialspoint.com/difference-between-simd-and-mimd. Accessed 16 October 2023.

[22] What is parallel computing? definition and faqs. HEAVY.AI. (n.d.). https://www.heavy.ai/technical-glossary/parallel-computing. Accessed 16 October 2023.

[23] Ginni. (2021c, July 30). What is the VLIW architecture?. Tutorials Point. https://www.tutorialspoint.com/what-is-the-vliw-architecture. Accessed 16 October 2023.

[24] Eckert Jr, J. P., Weiner, J. R., Welsh, H. F., & Mitchell, H. F. (1951, December). The UNIVAC system. In Papers and discussions presented

at the Dec. 10-12, 1951, joint AIEE-IRE computer conference: Review of electronic digital computers (pp. 6-16). Accessed 16 October 2023.
[25] Laptop Computer history. Computer Hope. (2023, July 13). https://www.computerhope.com/history/laptop.htm. Accessed 16 October 2023.