Computing, Information Systems, Development Informatics & Allied Research Journal
Vol. 12 No. 1, 2021 - www.cisdijournal.net

An Optimized Parallel Hybrid Architecture for Cryptocurrency Mining

Allenotor, D. & Oyemade, D. A.
Department of Computer Science
Federal University of Petroleum Resources Effurun (FUPRE)
Effurun, Delta State, Nigeria
E-mails: allenotor.david@fupre.edu.ng; oyemade.david@fupre.edu.ng
Phones: +234-8100534069, +234-8039209152

ABSTRACT

One of the requirements for mining cryptocurrency (Crypto) is that the secured ledger of the blockchain must be updated. However, updating the secured ledger requires that the miner develop and solve complex mathematical equations in higher orders hexadecimal 64-digit solution called a hash. In addition to this challenge, the mining processes of cryptocurrency are both resource and cost-intensive. The resources required include, but not limited to mining software, hardware, power (energy usage), CPU or compute cycles, and NP-hard problem. Apart from these numerous challenges that are associated with mining cryptocurrency, the amount of speed that is required to mine a single block is core. Cryptocurrency mining speed requirement is significantly important because only miners that can have the fastest mining device are most likely to get the reward (profit) from competing for a block. In this paper, we designed and implemented a model to speed up mining process which is capable of giving miners an advantage to arrive at a block earlier. The novelty of our architecture is that our design is based on high performance computing paradigm where we achieve processor speed up by parallelizing the number of processors p. We experimented by varying p = 4, 8, 16. Our experimental results where we used the MC6800 simulated on Easy68k emulator demonstrate feasibility of our proposed model and prove that speed was an essential key to cryptocurrency mining.

Keywords: Cryptocurrency, Blockchain, Architecture, Mining, Speedup, Bitcoin.

CISDI Journal Reference Format
Allenotor, D. & Oyemade, D. A. (2021): An Optimized Parallel Hybrid Architecture for Cryptocurrency Mining. Computing, Information Systems, Development Informatics & Allied Research Journal. Vol 12 No 1, Pp 95-104
DOI - https://doi.org/ 10.22624/AIMS/CISDI/V12N1P10. Available online at www.isteams.net/cisdijournal

1. INTRODUCTION/BACKGROUND

The year 2019 Coronal Virus Diseases (Covid'19) pandemic restructured the world-economy to embrace contactless associations in many spheres of human existence. Financial transactions were one of the most hit in the contactless scenarios. This has made the preference to become person-less mode of transacting business i.e., without any form of physical contact. In business terms, this paradigm shift could be described as an e-commerce system where exchange of goods and services is carried out using digital currency – a form of currency that exists only in digital or electronic form such that transactions can be carried out independent of the central bank (Weking et al., 2019). In a digital currency system, payments basically consist of a string of bits/digital records which could be copied and reused for other digital transactions. Since the string of bits could be used and reused. Chiuy in (Chiuy et al., 2018) described a situation where the digital token could be counterfeited and used twice which is referred to as the double-spending problem.

The problem of counterfeiting was mitigated by the introduction of a third party that monitors and manage transactions through a centralized ledger between a buyer's and seller's account (Haque and Rahman, 2020). An example of a third party is PayPal and users of this digital transaction trust the third party (Chiuy et al., 2018). Crypto mining is the creation of new units of cryptocurrencies by validating and generating transactions. These transactions carried out in a particular time frame are collected. Unlike fiat money where a centralized authority controls, monitors, and regulates the transaction processes, cryptocurrencies are decentralized. In the case of transactions using fiat money, the transactions requires that the banks put in place huge infrastructures such as banking structures, members of staff, etc., to monitor the transactions.

The requirement that the conventional banking system had to provide in other to offer banking services has many drawbacks over the Cryptocurrency system. A cryptocurrency transaction is the movement of digital coins from one digital wallet to another. When a transaction is carried out, the record of the transaction is shared with every participant on the network. To maintain transparency of this process, all the transaction details from the beginning of the process are recorded and stored in a distributed ledger called a blockchain. The miners process transactions by validating the true ownership of a currency from the sender to the receiver. Each transaction carries details of the hash of the previous transactions made by the owner, whereby the authenticity of a present transaction is done to verify it. Miners also prevent double-spending through the verification process.

Mining is carried out to generate and release new coins into the cryptocurrency system. Whenever a transaction is verified, miners collect the transaction details and add them to blocks that are being solved at that time. The process of providing a solution involves demystifying a set of complex mathematical puzzles. When a puzzle is completed, the block can be allowed to be added to the ledger and a token of a coin is given as a reward (which is a profit, in order words). Mining, therefore, is the process of competing to solve complex mathematical puzzles in return for a reward. Cryptocurrency makes use of cryptographic technology to ensure secured transactions of digital tokens on a decentralized network. Bitcoin for instance runs on powerful computers known as nodes which facilitate the transmission of information over the distributed network where a large volume of memory space and energy is consumed by the network of nodes (Narayanan et al., 2016). The identified network nodes propagate transactions which help to rapidly broadcast the transaction information across the network. The nodes that act as mining nodes on the network are known as miners.

The miner's group all unresolved transactions together as a block which is then appended to the blockchain. A complex mathematical puzzle called Proof of Work (PoW) which constitutes part of a bitcoin program is solved by each miner. Every miner finds a number known as nonce. The nonce is combined with the data in the block and then passed through the hash function. The prediction of the nonce which produces a result that falls within a certain range is considered an NP-hard problem. Aljabr in (Aljabr et al., 2019) gave a range of nonce value to fall within 0 to 4,294,967,296. The rest of this article is organized as follows. Section 2 presents the related work while in Section 3, we discuss the systematic methodology used. A proof of implementation is provided in Section 4 and our results are discussed in Section 5. Section 6 presents the conclusion and future work.

2. RELATED WORK

A large corpus of research efforts (Heller, 1978; Guy & Bruce, 2010; Salim, 1989) have studied the algorithm for the development of blockchain. In particular, a study was carried out on the design and development of a parallel PoW for permissionless blockchain system in (Shahriar, 2020) discussed on a network architecture, role of a manager and a reward system.

A. Network Architecture

To perform the PoW, some of the data used by the miners are identical, they include the block index, the hash value of the previous block, and the timestamp (Euromoney, 2020). However, the content of transactions and the nonce value chosen by the miners may differ. As the miner works separate, it may happen that multiple miners can use the same transaction data and nonce to create the next block. Since the miners do not share the data, they are using to find the cryptographic solution, there is no way to know that if they are using the same data. Again, as the miner competes with each other to create the same block and only one miner can be the successful miner (who find the solution first), the effort of all other miner become completely worthless. That decreases the scalability of the network and wastes a lot of energy. To get rid of this scenario, all miners can work parallel and no multiple miners do the same work. However, to achieve this, all miners will use the same transaction data but a different nonce. This means that all miners will use the same data except for the nonce for a certain block, thus ensuring that no multiple miners perform the same work.

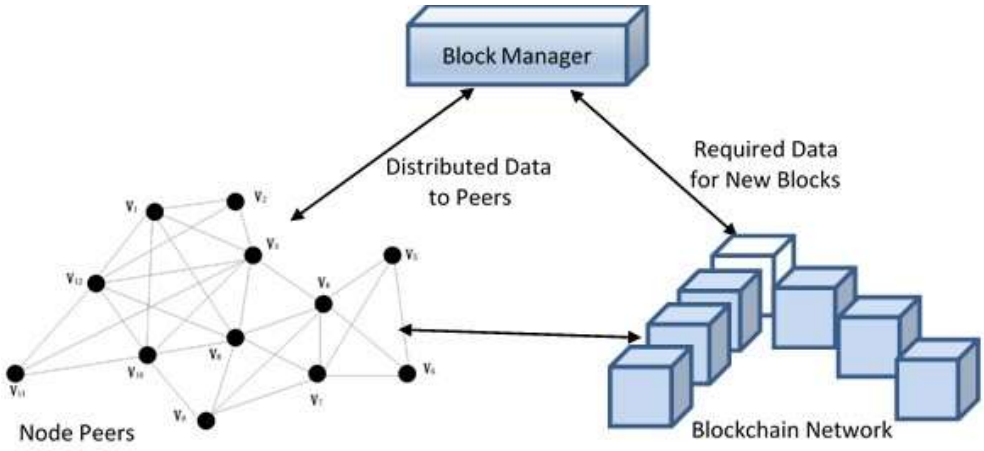

Figure 1. A Parallel Proof of Work (PoW) Network Architecture

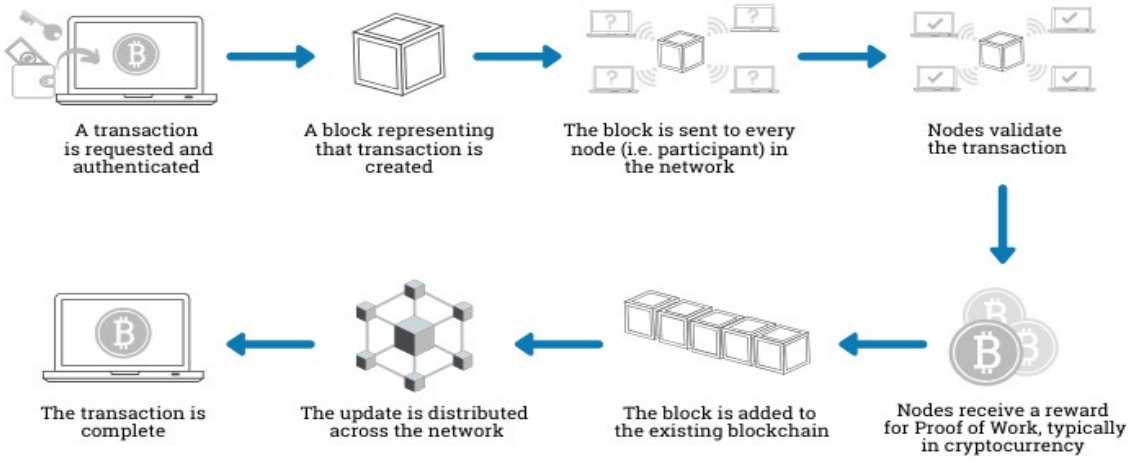

Figure 2. Transaction in Blockchain (Source: (Euromoney, 2020))

B. Role of a Manager

A manager is required to ensure that there are no two or more miners that use the same nonce value and that all miners use the same transaction data. The duplication can be checked in two chunks using linear comparison. The implementation prototype used a similar technique. The manager, who will be chosen from the miners, will be different in every epoch. Here, an epoch contains the time interval between two blocks. In this case, the manager rather than the miner will choose the nonce to compute. In the traditional way, every miner chose the transaction data and nonce value on their own. There should be a genesis block at the start of the Blockchain with no transactions. While a miner is randomly chosen as the manager for the next block, for the remainder of the blocks, the manager selected will be the one who solved the block before the previous block. All the miners will now compete with each other to solve the genesis block, following the traditional method. When the genesis block is solved by a miner, the epoch for the next block will begin. The proposed solution will be effective at this point.

C. Reward System

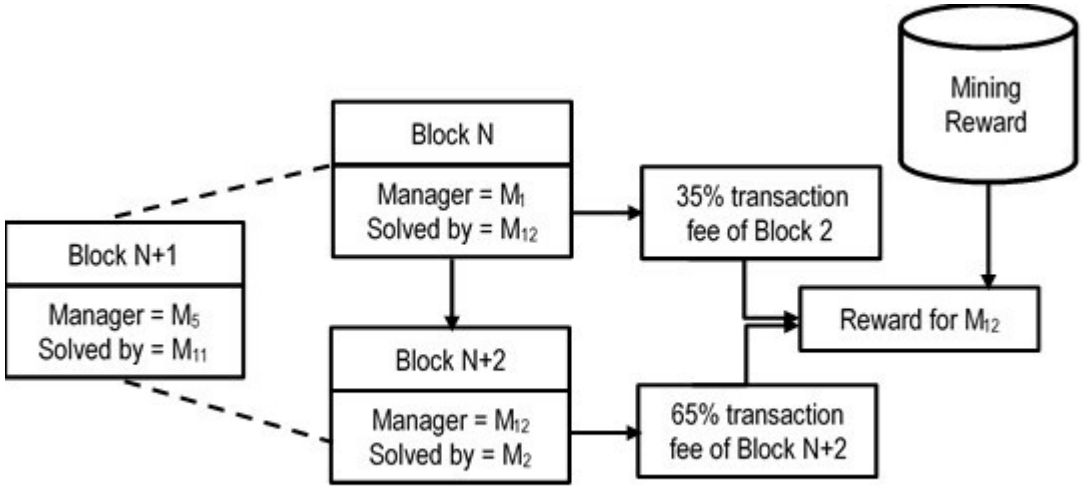

Figure 2. Proof of Work Proposed Reward System (Source: (Shahriar, 2020)).

In the reward system, the miners can mine a certain amount of cryptocurrency which in Bitcoin is 12.5 BTC for each block. In (Shahriar, 2020) proposed system, having created a block, the miner will be able to mine a certain amount of cryptocurrency, similar to the current system.

However, the miner will not receive all the transaction fees for all the transactions. Instead, the fees will be split with the manager, who will receive 65% of the transaction fee while the remaining 35% will be awarded to the miner who solved the block. Figure 2.0 shows an example. In this example, we illustrate the total reward (also called the transaction fees and mining currency) for a miner. For the given example, the reward is M12. That is, M12 solves block N. Thus, the miner gets the 35% transaction fees of all the transactions in block N. After that the miner will become the manager of block N+2. After solving the block N+2, the miner will get 65% transaction fees of that block. Additionally, the miner will get the corresponding mining reward to solve Block N.

3. METHODOLOGY

One of the approaches to achieve an optimized parallel architecture is by the development of a mixed mode version. One of the major aims of the mixed mode design is its interconnection network which provides an inter-processor communication. This mixed parallel mode performs its operation by executing instructions in Single-Instruction Multiple-Data (SIMD) or Multiple-Instruction Multiple-Data (MIMD) mode. The maximum performance can only be meant when it properly selects between SIMD and MIMD for every phase of the architecture. When switching between MIMD mode and SIMD, a processor cannot continue until all the processors gets to a switching point. The processor continues in MIMD mode without waiting instead of SIMD mode (Berg and Siegel, 1991). The mixed mode model has a similarity with SIMD model where the processors can either fetch instruction from their own memory or from instruction broadcast queue. When the system is executing instructions from their memory, the processor is performing on MIMD mode. Likewise, when the instructions are fetched from the broadcast queue it is in a SIMD mode (Berg and Siegel, 1991). Mixed mode parallel processing is a form of heterogeneous computing. One of the major aims of the mixed mode design interconnection network which provides an inter-processor communication.

i. **Single-Instruction Multiple-Data Mode**

The SIMD computer architecture or array processor is a parallel computer system that consists of N identical processors. Each identical processor uses a central Control Unit (CU) to carry out a single instruction. Similarly, the N processor is assumed to carry identical copies of a single program, where a local memory houses a copy of each processor. These processing elements are different from the sequential computers because it is unable to generate their own instructions but can receive the instructions from a global CU. The instructions could be as simple as adding two numbers or as complex as combining two list of numbers. Information can be encrypted in the instruction by telling a processor if it should be active and execute the instruction or idle and wait for the next instruction. Global clocking is the method used to implement a lock step operation i.e., a processor that is idle or inactive and does not complete an instruction will remain idle awaiting the next instruction. The time between the executions of two instructions is dependent on the executing instruction. In SIMD parallel computing, it is required that the processors communicate with each other during the execution processes for the purpose of exchanging data (Selim, 1989).

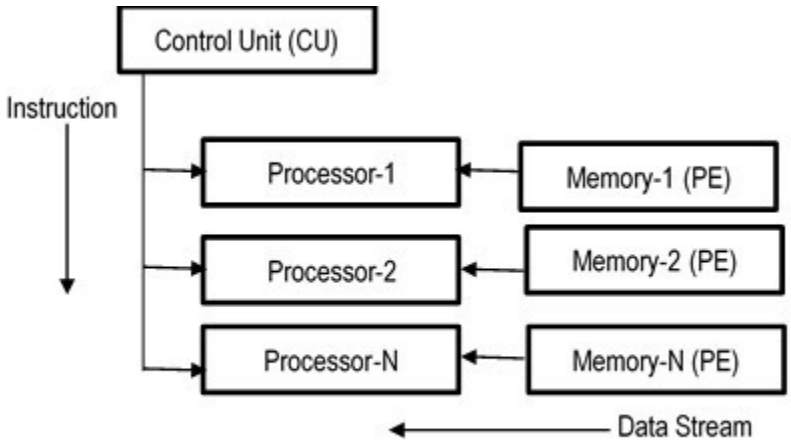

Figure 3. SIMD Architecture.

ii. **Multiple-Instruction Multiple-Data *mode***

The MIMD Architecture is a collection of N independent tightly coupled or loosely coupled processors with each processor having a dedicated memory and not directly accessible by other processors. MIMD computers have N processors, N streams of instructions, and N streams of data. In MIMD computers, each processors have its own control unit CU combined with its arithmetic and logic unit and local memory. The CU uses an instruction stream to control each processor. This will make each processor execute different programs on different data independently giving it an edge over SIMD computers. The processors in MIMD operates by sending data in one direction. Like SIMD which communicates either through shared memory or interconnection network, MIMD has either a tightly coupled machine that may share a common memory (multiple processors) or a loosely coupled machine that has an interconnection network (Multi computers) (Kaur and Kaur*, 2013*).

Communication of processors amongst themselves, helps algorithms use parallelism effectively. This occurs at interaction points, dividing the processes into sages. The speed of a processor is unpredictable where there is no certainty that an input required by a process will be produced in time by another. Kung in (Kung, 1982), suggested the use of two approaches for this problem. This makes them wait for inputs whenever necessary. This results in a synchronous algorithm and secondly by allowing processors to terminate or continue a process depending on the available information (the processor does not become idle), which leads to an asynchronous algorithm.

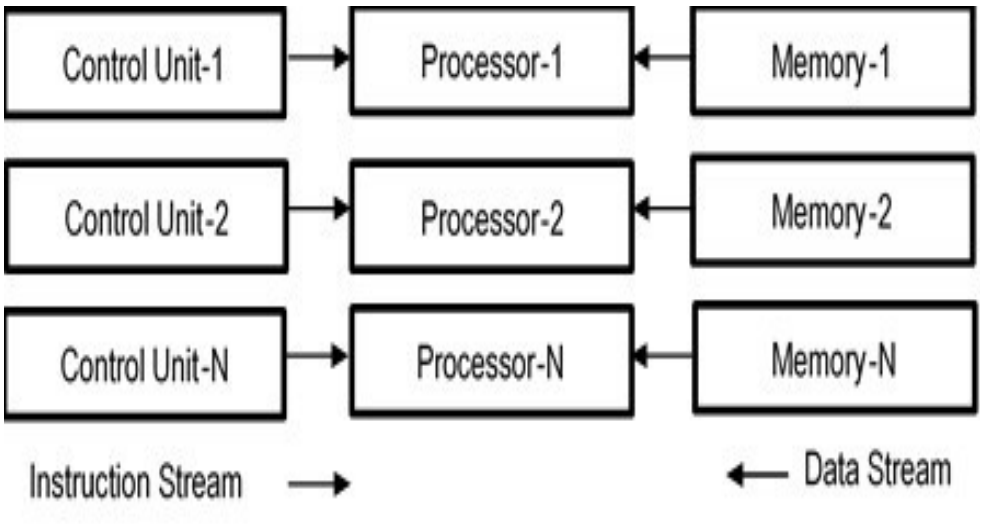

Figure 4. MIMD Architecture

A. Mixed Mode Operation

The mixed parallel mode performs its operation by executing instructions in SIMD or MIMD mode. The maximum performance can only be meant when it properly selects between SIMD and MIMD for every phase of the algorithm. When switching SIMD to MIMD, the cost should be considered. For switching between MIMD mode and SIMD, a processor cannot continue until all the processors gets to a switching point. The processor continues in MIMD mode without waiting instead of SIMD mode (Selim, 1989).

Table 1: Execution Time of SIMD, MIMD and Mixed Mode

Phase	Relative Execution Time(s)								
	SIMD			MIMD			Mixed mode		
	P_0	P_1	P_2	P_0	P_1	P_2	P_0	P_1	P_2
1	3	3	3	1	2	3	1	2	3
2	1	1	1	2	3	1	1	1	1
3	3	3	3	3	1	2	3	1	2

In the concepts of Siegel and Antonio (Siegel and Antonio, 1994), Table 1 shows three phases of the SIMD, MIMD and Mixed mode algorithm when tested with processors (P_0, P_1 and P_2). Each processor has different execution times with each phase non-uniform. From this, it shows that in mixed mode, the best section of modes will be MIMD for phase one, SIMD for phase two and MIMD for phase three.

4. IMPLEMENTATION

This section discusses the implementation technique for the proposed system on an MC68000 assembly language using an Easy68K simulator. The parallel algorithms SIMD, MIMD, and mixed-mode algorithm were executed on 4, 8, and 16 processors at different experimental runs simply by changing variables embedded in their data sections. In our first program, the SIMD process executes the loops and the control flow instructions in the MC68000. Arithmetic, data movement, and index calculation instructions are executed on the processing elements in SIMD mode. The processing element instruction stream is gotten through the MC68000's Fetch Unit Queue and is then executed synchronously on all processing elements. For example, in PASM (PArtitionable SIMD/MIMD) machine, the network seems to the processing elements as two memory locations, transmit and receive registers. Network transfers are made directly to the transfer registers using memory-to-memory move instructions. A great advantage of the SIMD version is due to the use of a FIFO queue in the Fetch Unit of the MC68000 (Jamieson *et al.,* 1987).

The second program was a MIMD program in which the MC68000 were only used for initiating the processing element programs. The processing elements executed all instructions asynchronously including all network, control flow, and arithmetic operations. The mixed mode was developed to take advantage of the fast barrier synchronization and the advantage of the execution time of MIMD programs. The processors fetch and execute instructions asynchronously with respect to one another. Rather than polling the network buffer, barrier synchronization was used to allow network operations to be carried out as simple memory-to-memory move operations as in the SIMD program MC68000 (Jamieson *et al.,* 1987). This lowered the amount of network overhead to a level comparable but slightly greater than the SIMD version due to the mode switching time. Another advantage of SIMD mode (i.e., faster instruction fetch and control flow instruction overlap) could not be realized in this version.

5. RESULTS AND DISCUSSIONS

Figure (a) shows the execution time vs. puzzle size observed in the parallel versions of the algorithm where p was set equal to 4 i.e., (p=4). The difference between the SIMD, MIMD and the mixed mode computation. From Figure 5, it was observed that SIMD and MIMD decreases as n increases. The only difference between these two modes is related to the contribution to the execution time of communication. Another aspect of the graph is the advantage of the SIMD version over the Mixed mode program. The difference is caused by the ability of the MC68000 to execute control flow in parallel with arithmetic.

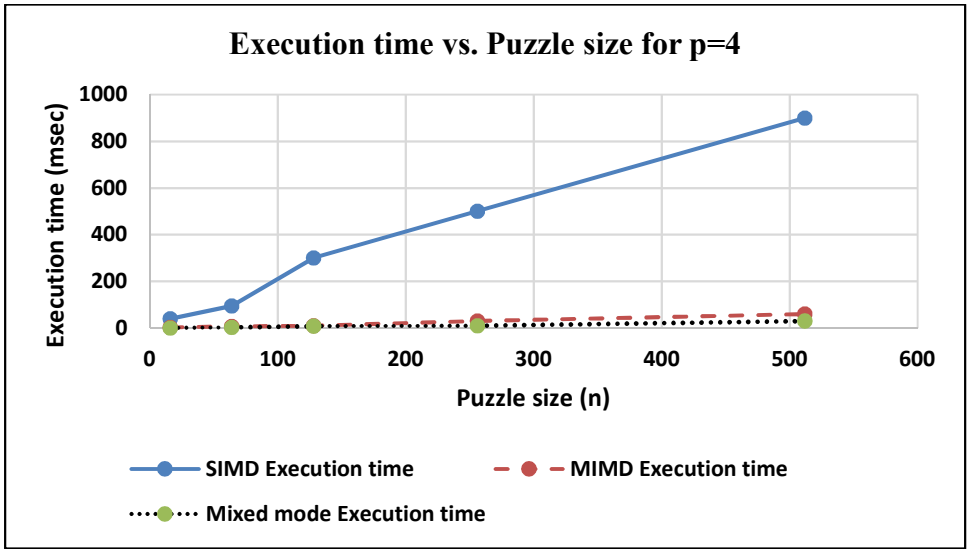

Figure 5. Execution time vs. Puzzle size (n) for p = 4

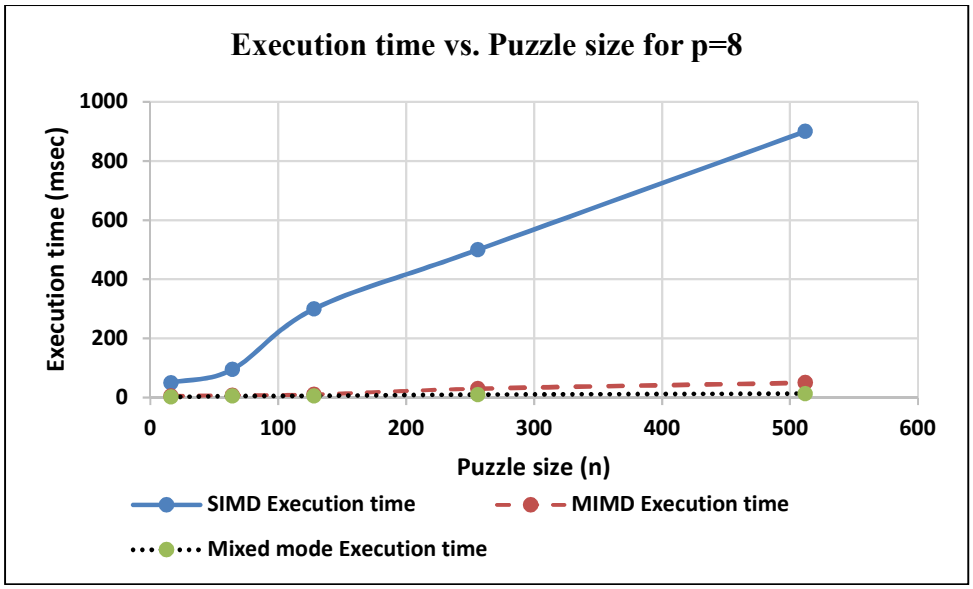

Figure 6. Execution time vs. Puzzle size (n) for p = 8

As seen in Figures 5, Figure 6, and Figure 7, the fastest parallel mode was clearly the mixed-mode version. After the mixed-mode versions, the MIMD version was next, with the SIMD version being the slowest. Note the similarity between Figure 5 and Figure 6. This is due to the time complexity of the algorithm, which is proportional to (N/P) p (p + 1)/2. This complexity is the same for P = 4 and P = 8 (N/ P) p (p+ 1)/2 = (N/4) (2 * 3)/2 = (N/8) (3 * 4)/2). However, a slight difference exists due to variations in overheads and control flow overlap. The mixed mode version was faster than the other versions and improved in relation to the other versions as N increased.

Its speed can be attributed to it possessing almost no overhead for network transfers and the added advantage of the control flow overlap when in SIMD mode. In fact, the only overhead present in this version but not present in the others was that of explicitly transferring control between SIMD and MIMD modes. Note that as N increases the advantage of mixed the mode over the MIMD mode improves due to the (N/P) p (p + 1)/2 complexity.

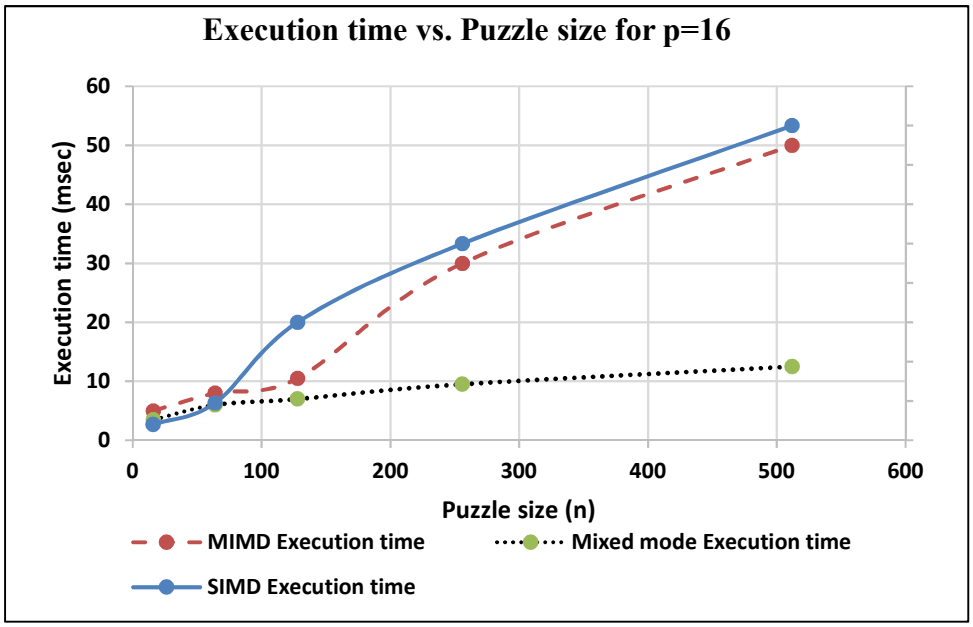

Figure 7. Execution time vs. Puzzle size (n) for p = 16

The operational benefits of SIMD control flow overlap and SIMD synchronized network transfers. Also, the mode-changing overhead of the mixed-mode is outside of the loop and occurs only p (p + 1)/Z times. There is a large difference is evident between the SIMD mode and the other two modes.

6. CONCLUSION

We have studied the impact of SIMD, MIMD, and mixed-mode parallelism for the implementation of cryptocurrency mining. Using the MC6800, we simulated these three architectures on the Easy68k emulator and demonstrated the feasibility of our model. We have proven that speed is required as an essential key to cryptocurrency mining. Hypothetically, we have used the concepts of high-performance computing to connect processors (processing elements/processing nodes) together in three different architectures; MIMD, SIMD, and the hybrid of MIMD and SIMD called the mixed mode. We demonstrated that for pk where $k \geq 2$ the speed with which a miner solves a given number of puzzles n was inversely proportional to the execution time. It therefore means that with our hybridized approach, lesser resources are required to solve a higher number of puzzles which earns more profit for the miner.

7. FUTURE WORKS

Future work includes simulation/experiments using a live data to get an in-depth analysis of how cryptocurrency mining could dynamically restructure their resources using a self-resource allocation scheme. Another important direction to this research includes the development of an aging algorithm to help minimize the influence of aged data.

REFERENCE

1. Aljabr, A., Sharma, A., & Kumar, K. (2019). Mining Process in Cryptocurrency Using Blockchain Technology: Bitcoin as a Case Study. Journal Of Computational and https://doi.org/10.1166/jctn.2019.8515
2. Theoretical Nanoscience, 16(10), 4293-4298.
3. Euromoney. (2020). Blockchain Explained: How does a transaction get into the blockchain? Euromoney.com. Retrieved 4 February 2022, from https://www.euromoney.com/learning/blockchain-explained/how-transactions-get-into-the-blockchain.
4. Haque, A., & Rahman, M. (2020). Blockchain Technology: Methodology, Application and Security Issues. International Journal of Computer Science and Network Security, 20(2), 21-30. Retrieved 4 Sept 2021, from https://www.researchgate.net/publication/347881071_Blockchain_Technology_Methodology_Application_and_Security_Issues.
5. Heller D. (1978). A Survey of parallel Algorithm in Numerical Linear Algebra, (SIAM Rev. 20).
6. Howard Jay Siegel and John K. Antonio (1994) Views of Mixed-Mode Computing and Network Evaluation: International Symposium on Parallel Architecture, Algorithms and Networks.
7. Jamieson, L. H., Gannon, D. B., and Douglass, R. J. (Ms.). (1987). The Characteristics of Parallel Algorithms. MIT Press, Cambridge, MA, 1987, pp. 65-100.
8. Kung, H.T., (1982). Notes On VLSI Computation, in Parallel Processing System. Cambridge University Press, Cambridge.
9. Mandeep Kaur and Rajdeep Kaur. (2013) A Comparative Analysis of SIMD and MIMD Architecture: international Journal for Advanced Research in Computer Science and Software Engineering, (Volume 3) Pp. 1151 & 1154.
10. Narayanan, A., Bonneau, J., Felten, E. W., Miller, A., Goldfeder, S. (2016). Bitcoin and Cryptocurrency Technologies - A Comprehensive Introduction. Princeton University Press. ISBN: 978-0-691-17169-2
11. Selim G. Aki. (1989) The Design and Analysis of Parallel Algorithms", Pp 17, 18 & 29.
12. Shahriar S. S. and Qusay H. M. (2020). Improving Transaction Speed and Scalability of Blockchain Systems via Parallel Proof of Work. *Future Internet* 12, no. 8: 125. https://doi.org/10.3390/fi12080125
13. Thomas B. Berg and Howard Jay Siegel. (1991) Instruction Execution Trade-Offs for SIMD vs. MIMD vs. Mixed Mode Parallelism.
14. Weking, J., Mandalenakis, M., Hein, A., Hermes, S., Bohm, M. and Krcmar, H., 2019. The Impact of Blockchain Technology on Business Models – A Taxonomy and Archetypal Patterns. Electronic Markets, 30(2), pp.285-305.