PAKSUPERCOMPUTER: AN OPEN-SOURCE, SCALABLE, AND HETEROGENEOUS SUPERCOMPUTING PLATFORM

Tassadaq Hussain[1,2], *Amna Haider* [2]

[1] *Centre for AI and Big Data Namal University Mianwali, Mianwali, Pakistan*
[2] *Pakistan Supercomputing and Barcelona Supercomputing Center Spain*

tassadaq@ucerd.com

Keywords: HPC, SUPERCOMPUTING, DISTRIBUTED COMPUTING, HETEROGENEOUS COMPUTING, HARDWARE ACCELERATION, PARALLEL PROGRAMMING

Abstract

The recent advances in large vision and language models, along with generative artificial intelligence, have led to a growing demand for compute resources such as increased processing capacity, faster memory, and larger storage. In the current scenario, supercomputing is the only domain capable of solving complex and compute-intensive algorithms for large datasets. Thus, supercomputers are considered the backbone of all fields of academia, research, innovation, and commercialization. In this work, we propose and develop a Scalable Heterogeneous Supercomputer called PakSupercomputer. The PakSupercomputer uses heterogeneous processing cores, including multiple CPUs, Tensor Processing Units (TPUs), and Field Programmable Gate Arrays (FPGAs). An Elastic Parallel Programming framework is deployed to provide an easy-to-use programming environment. This programming environment assists high-performance computing developers in writing applications without dealing with the complexities of heterogeneous distributed hardware architectures. The results confirm that when testing benchmarks with larger-scale, complex problems and cloud computing applications, the PakSupercomputer proves to be a low-power, Quadrillion Floating-Point Operations Per Second (FLOPS) supercomputer.

1 Introduction

The processor industry has followed Gordon Moore's prediction [1], which guided performance improvements in High-Performance Computing (HPC) systems via increasing clock frequencies in processor architectures. However, due to the constraints of the power wall [2], the industry shifted its focus from single-core to multi-core processor designs. This transition introduced a new era in HPC, characterized by on-chip parallel processing techniques. Modern HPC architectures now incorporate thousands of heterogeneous multi-cores, creating new challenges, such as programmability, power consumption, and scalability. Initially, Floating-Point Operations Per Second (FLOPS) served as the sole metric for assessing supercomputer performance. However, today, the evaluation of HPC systems considers not only raw computational power but also scalability and energy efficiency, expressed as FLOPS per watt. A summary of the most energy-efficient supercomputers, according to the Green500 list, is provided in Table 1.

High-Performance Computing (HPC) is a well-established field dedicated to processing complex, compute-intensive applications. Recently, the field has experienced a rapid surge in technological advancements, particularly concerning computational capabilities and energy efficiency. Leading global technology companies such as Google, Facebook, and Amazon have also prioritized power efficiency. For instance, Facebook has achieved a 38% reduction in power consumption in its data center servers through improvements in power distribution and cooling systems. The latest Top500 supercomputers list [3] demonstrates exponential growth in supercomputer performance, with tenfold improvements observed every 3.6 years. The current leading supercomputer delivers 62.68 PFLOPS while consuming 15 megawatts (MW) of power. Following this trend, achieving exascale performance by 2025 is projected to require 400 MW of power. However, HPC architects aim to achieve this milestone within a power budget of 20 MW, targeting an energy efficiency of 50 GFLOPS/W.

To meet these energy and performance targets, significant advancements in processor architectures, programming models, data storage systems, networking technologies, and cooling methods are required. The interdependence of performance, cost, and energy consumption is critical when designing computer architectures, whether for embedded systems, cloud platforms, or bare-metal HPC systems. Consequently, the design of a computer system architecture must emphasize energy efficiency, rapid processing capabilities, and the effective execution of tasks to handle large-scale datasets. With recent advancements in cloud computing, data centers, and edge computing, HPC technologies are undergoing constant evolution. These technologies enhance national security, promote research and development, and facilitate the timely production of advanced industrial equipment. As such, HPC is a key indicator of a country's technological prowess and economic strength.

Table 1: Top 5 Energy-Efficient Supercomputers from the Green500 List

Name	System Type	GFLOPS/W
MN-3	Fujitsu A64FX	39.38
Frontier TDS	AMD Epyc, MI250X GPU	62.68
LUMI-C	AMD Epyc, MI250X GPU	59.92
Adastra	AMD Epyc, MI250X GPU	58.12
Fugaku	ARM-based Fujitsu A64FX	28.33

In this work, we propose a Scalable Heterogeneous Supercomputer, referred to as PakSupercomputer. The PakSupercomputer consists of three primary subsystems: a) Heterogeneous Processing Cores, b) Open-Source Distributed System Stack and c) Elastic Programming Model. The heterogeneous processing subsystem integrates multiple processing cores, Tensor Processing Units (TPUs), and Field Programmable Gate Array (FPGA) accelerators. The HPC nodes are interconnected using a high-speed, low-latency switch, ensuring high throughput and scalability. Multi-node HPC clusters are configured with open-source distributed systems and libraries. To simplify application development, an Elastic Parallel Programming Model is employed. This model abstracts the complexity of heterogeneous and distributed hardware architectures, enabling developers to focus on their applications. The results demonstrate that PakSupercomputer is a low-power supercomputing solution capable of achieving Quadrillion Floating-Point Operations Per Second (FLOPS). It is designed to address large-scale, complex problems, including distributed artificial intelligence (AI) and cloud applications. By utilizing open-source technologies for heterogeneous data processing, simulations, and AI-driven solutions, PakSupercomputer aims to address local challenges in healthcare, education, and industry. The system provides indigenous, cost-effective solutions tailored to the needs of the local community.

2 Related Work

This section gives an overview of research that is already relevant to our study.

Recent investigations have explored the prospect of field-programmable gate arrays (FPGAs) in high-performance computing (HPC). Craven et al. [4] performed a comprehensive evaluation of FPGAs and general processors, focused on floating-point performance and acquisition costs. In his study, he highlighted the potential fiscal challenges in utilizing FPGAs for HPC applications.

In another study by Tian et al., [5] a densely parallelized Quasi-Monte Carlo simulation engine on an FPGA-based supercomputer, named Maxwell, was developed at the University of Edinburgh. Remarkably, under specific conditions, their hardware implementation outperformed software implementations on Xeon processors by a factor of 10^3.

Rajovic et al. [6] analyzed performance in mobile System-on-Chip architecture advancements since 1990. Their research involved setting up a chip cluster to evaluate network scalability and assess performance and productivity in real-world application scenarios.

The authors also presented the first large cluster built from ARM multicore chips; the Tibidabo architecture [7]. They uncovered key design principles to boost high-performance computing (HPC) system efficiency through an in-depth analysis of performance and energy usage. Simulations showed that a 16-core ARM Cortex-A15 chip cluster achieves remarkable energy efficiency (1046 MFLOPS/W) and speeds up processing by 8.7 times.

Where recent innovations in the financial computation field indicate that FPGAs hold significant importance for the Black–Scholes and Monte Carlo models, specifically for option pricing. From the review of 99 research papers including their work O Mahony et al. [8] gives an insight into how the use of FPGAs provides a high degree of performance improvement of between 270 and 5400 times than in the traditional CPU implementations and energy efficiency. However, the authors note that there are still shortcomings in the FPGA design, as well as the complexity of programming, which makes their use a big issue to date. Nevertheless, it is essential to highlight the fact that further research opportunity exists in the choice of High-Level Synthesis as the tools should be able to simplify the programming of FPGAs in the future. This line of work adds further support to the increasing adoption of FPGAs in finance-related solutions, as well as their potential for transforming the field of high-performance computing in the field.

Kondo et al. [9] proposed an extension to the HykSort algorithm to leverage GPU acceleration for large-scale distributed sorting on heterogeneous systems. Their approach offloads computationally intensive phases to GPUs and employs an iterative strategy to address GPU memory limitations. Evaluations on the TSUBAME2.5 supercomputer demonstrated significant speedups for large datasets. However, the study highlighted the bottleneck of CPU-GPU communication, suggesting that future advancements in GPU technology are crucial to fully realize the potential of GPU-accelerated sorting.

Milojicic et al. [10] discussed the increasing need for heterogeneous computing in HPC systems to address the limitations of traditional homogeneous architectures. They argue that a diverse hardware stack, including specialized accelerators like GPUs and FPGAs, is essential to achieve optimal performance and energy efficiency. The paper explores the impact of heterogeneity on various aspects of HPC systems, including interconnects, memory models, power and cooling, and application domains. It highlights the challenges and opportunities associated with managing and programming heterogeneous systems, emphasizing the importance of flexible and adaptable software frameworks.

Riedel et al. [11] explored the use of heterogeneous modular supercomputing architectures (MSAs) for accelerating machine learning (ML) workloads. They leveraged the modularity of MSAs to efficiently utilize both CPUs and GPUs, demonstrating significant performance improvements for large-scale ML tasks in remote sensing and health sciences. Their work highlights the potential of MSAs in addressing the growing computational demands of modern ML applications.

Moreover, Chang et al. [12] talked about the rise of big data analysis, shedding light on new opportunities for research and the impact on social science phenomenon studies it brings forward. They made a compelling case for why theoretical foundations still matter and pointed the way for future research.

Tassadaq et al. proposed a wide range of computer architectures including heterogeneous processing core [13–16], bus scheduler [17, 18, 18, 19], memory controller [20, 21] and memory system for high-performance computing and parallel processing. They developed an FPGA-based supercomputer using Xilinx Zynq SoCs [22] and designed RISC-V virtual cluster [23]. Their work also introduces novel memory systems and virtualization techniques that enable the execution of multiple tasks in parallel across heterogeneous processing cores, effectively reducing the processor-memory performance gap.

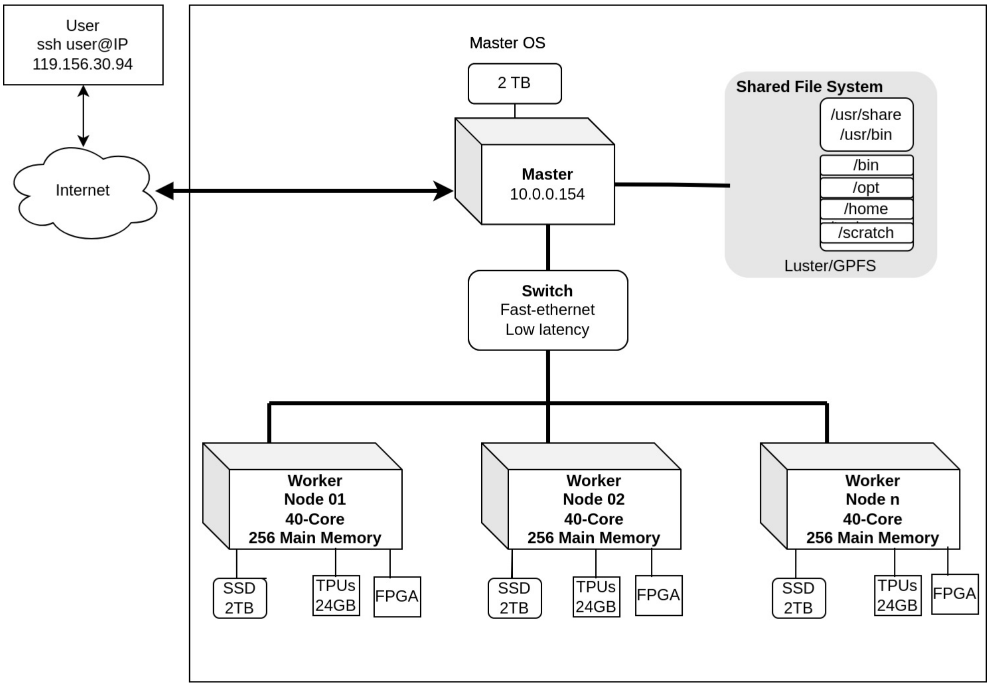


Figure 1: PakSupercomptuer: Master/Worker Node, Networking, and File System Configuration

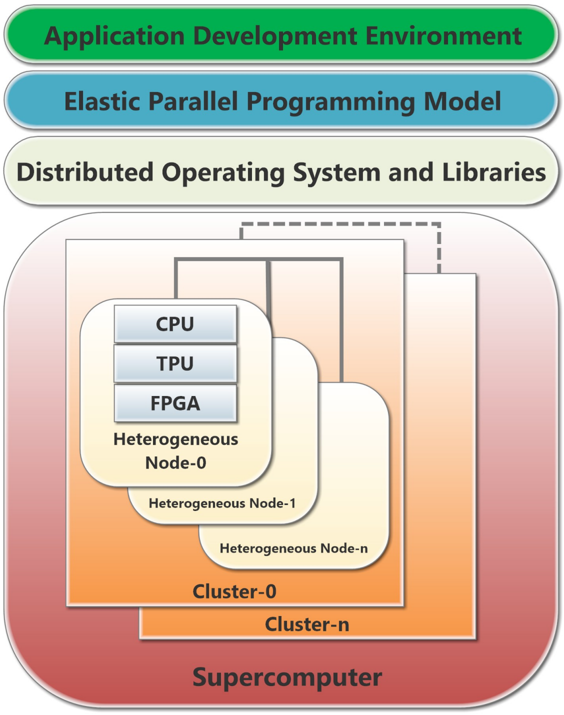

Application Development Environment
Elastic Parallel Programming Model
Distributed Operating System and Libraries
CPU
TPU
FPGA
Heterogeneous
Node-0
Heterogeneous Node-1
Heterogeneous Node-n
Cluster-0
Cluster-n
Supercomputer

Figure 2: PakSupercomputer: Scalable Heterogeneous Supercomputer System Architecture

3 Proposed Scalable Heterogeneous Supercomputer (PakSupercomputer)

In this section, we describe the proposed Scalable Heterogeneous Supercomputer (PakSupercomputer) architecture (Figures 1 and 2), and software stack (Figure 3). The architecture is organized into four key subsections: *Hardware System*, *Distributed System Software Stack*, *System Configuration*, and *Elastic Programming Models*.

3.1 Hardware System

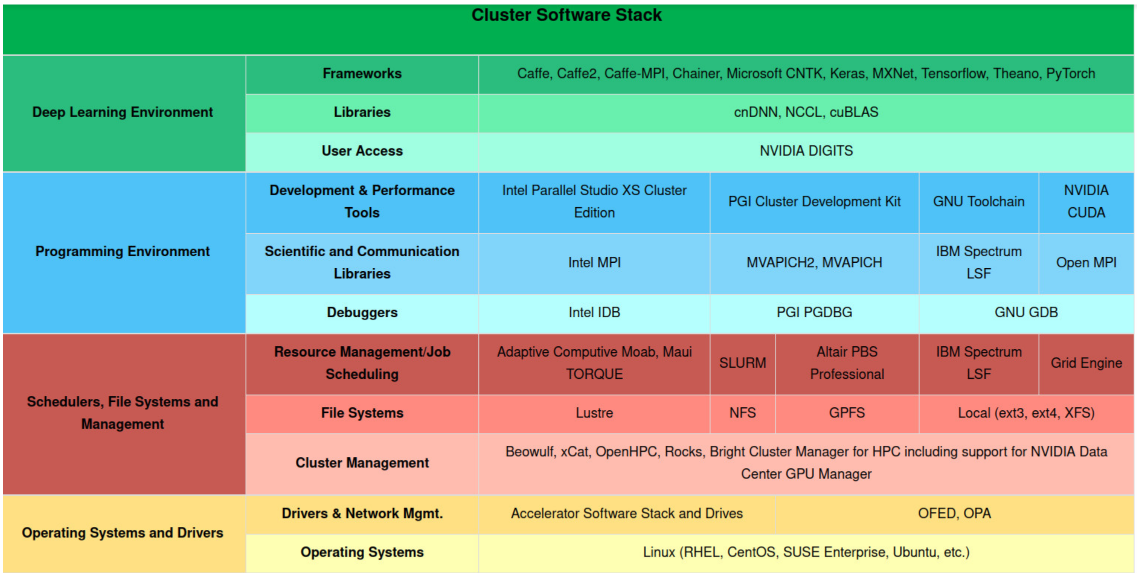
The *Hardware System* consists of a heterogeneous processing architecture with a master node and multiple scalable worker nodes shown in Figure 1. The nodes are connected through a high-speed switch to allow fast data transfer and communication throughout the cluster. The hardware system architecture is further subdivided into a) Multi-core System, b) Network Attached Storage, c) Scalable Switch, and d) Rack System.

3.1.1 Multi-Core System

Every multi-core system referred to as a compute node has a powerful x86-based CISC architecture. Each node comes with 128GB LPDDR4 RAM, 2TB local storage, and an Nvidia RTX 4070 GPU for faster computation. The master node also has an FPGA connected through a PCIe slot, which offers hardware acceleration for certain tasks that need high parallelism and low-latency processing. This FPGA, in conjunction with the GPU, transfers demanding computational tasks from the CPU to optimize the use of computational resources.

3.1.2 Network Attached Storage

The system uses a centralized Network Attached Storage (NAS) with NFS (Network File System) to offer shared storage across every node. The shared storage uses a solid-state disk for seamless data access within the cluster. The SSDs provide faster read/write speeds reducing latency and improving the overall performance of data-intensive operations within the cluster. The LSI MegaRAID 9361-8i RAID controller is used which supports eight SATA or SAS drives, to handle 10TB of NAS with high performance and reliability. The MegaRAID 9361-8i offers hardware RAID configurations including RAID 0, 1, 5, 6, 10, 50, and 60, which provides flexible options for data redundancy and performance optimization. The controller



Cluster Software Stack						
Deep Learning Environment	Frameworks	Caffe, Caffe2, Caffe-MPI, Chainer, Microsoft CNTK, Keras, MXNet, Tensorflow, Theano, PyTorch				
	Libraries	cnDNN, NCCL, cuBLAS				
	User Access	NVIDIA DIGITS				
Programming Environment	Development & Performance Tools	Intel Parallel Studio XS Cluster Edition	PGI Cluster Development Kit		GNU Toolchain	NVIDIA CUDA
	Scientific and Communication Libraries	Intel MPI	MVAPICH2, MVAPICH		IBM Spectrum LSF	Open MPI
	Debuggers	Intel IDB	PGI PGDBG		GNU GDB	
Schedulers, File Systems and Management	Resource Management/Job Scheduling	Adaptive Computive Moab, Maui TORQUE	SLURM	Altair PBS Professional	IBM Spectrum LSF	Grid Engine
	File Systems	Lustre	NFS	GPFS	Local (ext3, ext4, XFS)	
	Cluster Management	Beowulf, xCat, OpenHPC, Rocks, Bright Cluster Manager for HPC including support for NVIDIA Data Center GPU Manager				
Operating Systems and Drivers	Drivers & Network Mgmt.	Accelerator Software Stack and Drives		OFED, OPA		
	Operating Systems	Linux (RHEL, CentOS, SUSE Enterprise, Ubuntu, etc.)				

Figure 3: Supercomputing Software Stack for Parallel and Distributed Application Development

has integrated 1GB cache and optional battery backup, that ensures data protection.

3.1.3 Scalable Switch

A Cisco Nexus 6001 switch is used for networking, offering a maximum of 48 x 10GbE ports and accommodating various combinations of 10GbE/40GbE connections. It has a switching capacity of 1.28 Tbps, guaranteeing fast data transfers with minimal delay among nodes. The switch enhances inter-node communication efficiency by supporting RDMA over Converged Ethernet (RoCE), eliminating the need for CPU involvement in data transmission. The switch is configured with features such as link aggregation and Quality of Service (QoS) settings which prioritize the critical traffic, ensuring that high-priority tasks along with required bandwidth. Multiple power supplies are integrated for dual-homing to enhance the overall reliability and fault tolerance of the network. The network strategy improved the supercomputer system's efficiency and improved its scalability and responsiveness in processing complex computational tasks.

3.1.4 Rack System

The entire cluster is housed in a rack system designed to efficiently manage thermal and power requirements while optimizing space utilization. Each node within the rack is equipped with independent cooling mechanisms to maintain system stability during high workloads.

3.2 Distributed System Software Stack

The distributed system uses a multi-layered software stack, as shown in Figure 3, to manage and execute large-scale computations across multiple nodes. The stack guarantees effective and scalable data management, task scheduling, resource allocation, and security. The next parts will explain the main elements of the software stack: a) Operating System, b) Networking, c) Distributed File System, d) Management and Monitoring, and e) Scheduling and Virtualization.

3.2.1 Operating System

To deploy a Linux operating system that manages the heterogeneous computing nodes. Rocky Linux Server 9.4, is used that offers extensive support for heterogeneous architectures, including CPUs, GPUs, and FPGAs. The OS is optimized for space sharing, power efficiency, and virtualization. This configuration allows for the seamless integration of diverse hardware accelerators and efficient handling of parallel workloads, making it suitable for large-scale supercomputing applications. The master node manages the shared storage, task scheduling, and hardware resource management, ensuring that application tasks are efficiently distributed among the client nodes. By leveraging a high-speed local area network (LAN), we can achieve low-latency communication, which is critical for parallel processing and real-time data sharing. Additionally, implementing protocols such as MPI (Message Passing Interface) enhances inter-node communication, facilitating seamless collaboration among the nodes during computational tasks. This configuration not only boosts performance but also enhances the overall scalability of our supercomputing environment.

3.2.2 Distributed File System

The Network File System (NFS) is utilized to enable distributed storage, enabling efficient access to shared data by all nodes. The distributed file system uses Lustre and GlusterFS to reduce the metadata server and object storage targets to be configured, whereas the GlusterFS can be set up with multiple storage nodes for redundancy and scalability. This method guarantees that important datasets and application files can be accessed throughout the cluster without needing to be locally duplicated on every node.

The PakSupercomputer distributed file system also shares folders between users and nodes, which enhances collaboration and performance. The system shared file system ensures that all users have seamless access to essential resources, which enhances the efficiency of research and computational tasks. Following are the selected shared directories and information:

- $/usr/share$: folder provides access to shared data files, configuration files, and documentation, essential for maintaining consistency across different nodes.
- $/opt$ is used for installing optional application software, allowing users to run a variety of applications without compatibility issues.
- $/home$ facilitates individual user data storage, ensuring that personal research files are easily accessible.
- $/usr/binand/bin$ contain essential executables and user commands necessary for system operations and application functionality.

By sharing these directories, the supercomputer not only improves the collaborative potential but also mitigates issues related to file access and version control, ultimately leading to enhanced productivity and streamlined workflows.

3.2.3 Networking

The networking setup uses DHCP to automatically allocate IP addresses to all devices on the network. The master node can securely communicate with worker nodes by using public key authentication, as user accounts are consistently established on all nodes. All worker nodes have access to a shared directory on the master node through NFS, ensuring that file access in the cluster is both consistent and synchronized.

The PakSupercomptuer has implemented a robust networking strategy for effective communication between the master and worker nodes. The master node is assigned a Global IP address of $119.156.30.94$. The global IP makes the supercomputer accessible from external networks for administrative and access purposes using secure shell protocol SSH. While the client nodes are configured with local IP addresses to optimize internal communication. The dual-IP strategy allows for secure and efficient data transfer within the local network while maintaining external connectivity for system management and monitoring.

3.2.4 Management and Monitoring

A system for managing modules has been put in place to load and unload computational modules as needed for tasks. This system enables the allocation of resources as needed, improving efficiency by effectively utilizing both the GPUs and the FPGA for targeted workloads. Monitoring tools are utilized to observe the wellness of the system, the usage of resources, and the effectiveness to guarantee high availability and reliability throughout the cluster. The PakSupercomputer uses a Lmod module management system that streamlines access to software packages and their dependencies. The Lmod allows users to dynamically load and unload specific versions of applications, libraries, and tools based on the project requirements. It provides a flexible and efficient method to handle diverse and complex software environments and avoid conflicts between different versions of software at runtime. The modular approach of Lmode enhances productivity by allowing users to effortlessly configure their environments, ensuring easy integration of tools like MPI, compilers, and libraries tailored to their specific workloads.

3.2.5 Scheduling and Virtualization

The scheduling system is done by SLURM (Simple Linux Utility for Resource Management), which supports job scheduling and resource allocation in a distributed environment. SLURM schedules the computational tasks over the distributed nodes, taking into account the availability of hardware accelerators (GPUs, FPGAs). Virtualization tools are also integrated to provide flexibility and scalability, enabling isolated execution environments for different applications and tasks. KVM is used to provide a full virtualization environment, along with docker which provides lightweight containerization. KVM utilizes a hypervisor whereas Docker creates the container images and manages networks.

3.3 Development Framework and Libraries

The system supports various programming models that bridge the gap between hardware and software. These include MPI (Message Passing Interface), OpenMP (Open Multi-Processing), OpenCL (Open Computing Language), and OpenACC. Each model offers distinct advantages for different types of parallel applications, ensuring that the system can accommodate a wide range of computational tasks. The development framework and libraries are optimized for heterogeneous computing, supporting both GPU and FPGA acceleration where applicable. The complete software application development stack is shown in Figure 3.

Parallel and distributed computing present state-of-the-art contributions in terms of a wide range of programming models and libraries prepared for achieving high computational productivity across different levels of hardware heterogeneities. Out of the first framework, MPI/LAM was very influential in the further development of message-passing interfaces. Initially authored as a module of Open MPI, LAM stands for Local Area Multicomputer which was designed to supply an infrastructure that permits the heterogeneity of joined networks of computers to act as one parallel computing framework. With this system, there was the ability to orchestrate clusters and it supported more than one form of communication such as shared memory, TCP/IP, Myrinet, and Infiniband. These features enabled low latency, high bandwidth client-to-client communications implying that LAM/MPI is well suited for distributed computing in HPC. Open MPI came as the successor to LAM adding more modularity and accommodating multiple platforms and interconnects to today's computational world.

For Streaming Multi-Processing (SMP), OpenMP (Open Multi-Processing) API is employed for C, C++, and Fortran parallel application offload execution. The OpenMP uses a dynamic fork-join model of parallelism, where it is relatively easy to allow developers to parallelize the code using only pragmas or directives. It abstracts thread management of fine-grained, dynamic workloads and synchronization and data environment, which makes it suitable for Multicore processors. MPI for distributed memory and OpenMP for shared memory information are combined in hybrid programming models, to enable applications to migrate to multiple core and many node systems seamlessly.

OpenCL (Open Computing Language) takes parallelism even further by allowing for what is known as heterogeneous

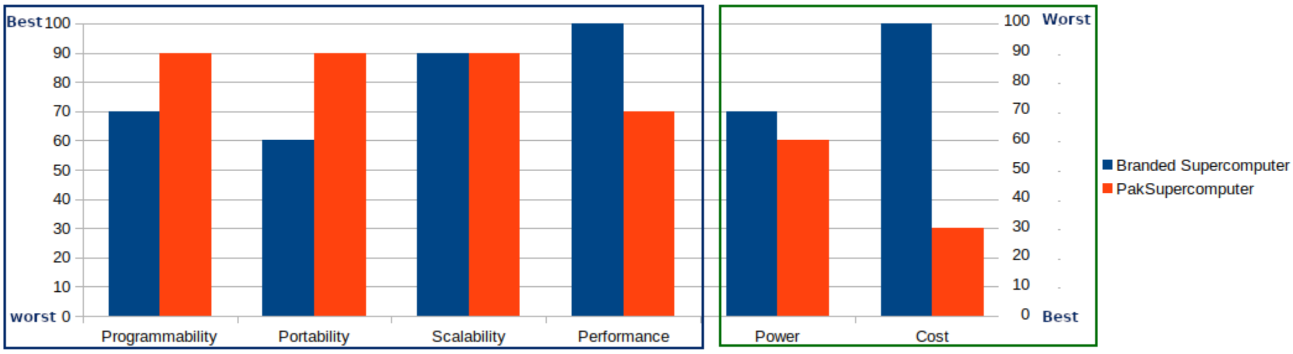


Figure 4: Efficacy of PakSupercomputer against Programmability, Portability, Scalability and Performance (Higher is Better), Power and Cost Lower is Better

programming. It supplies a single API platform that enables developers to write software that executes on CPUs, GPUs, FPGAs, and even other accelerators. This abstraction allows optimization of the compiled code in order to run it across a wide range of hardware platforms without having to rewrite the code. OpenCL has device memory separately from the host, it has kernels that run across different platforms and uses explicit memory transfers for optimization purposes. In the same context, CUDA, which is accelerated by NVCC (NVIDIA CUDA Compiler), is capitalizing on NVIDIA GPUs' high parallel composition capability. NVCC enables the code compilation of highly optimized GPU kernels especially in the application of intense computation for instance deep learning and real-time video analytics.

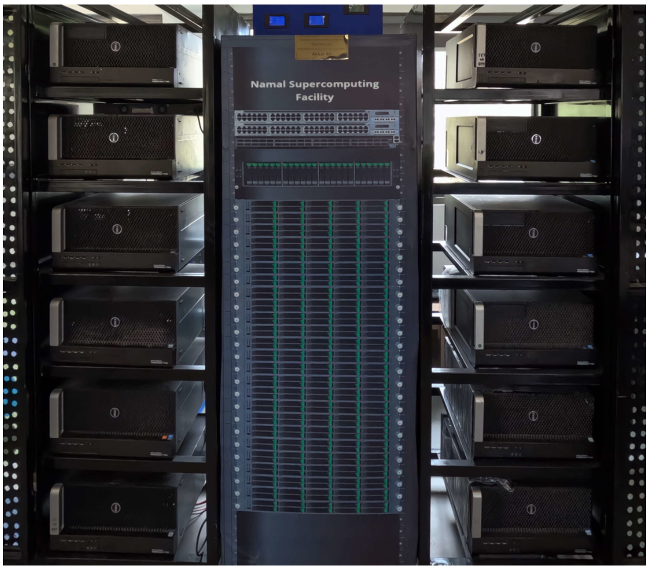


Figure 5: Photograph of PakSupercomputer deployed at the Namal Centre for AI and Big Data

OpenACC is a rapidly growing API that aims at simplifying the process of accelerating programs with the help of GPUs. Unlike OpenCL or CUDA, OpenACC employs pragmas, and the developer is to identify what part of the code has to be accelerated, or executed on a GPU. This approach hides the hardware complexity and generates memory transfers that can be easily optimized for both multi-core CPUs and GPUs. OpenACC coupled with NVCC for GPU execution provides a good set of tools for total throughput, and low latency applications. It is most useful where an iterative computing process such as prototyping or optimizing the performance of a program is needed.

The combinations of these frameworks are OpenMPI for distributed memory parallelism, OpenMP for shared memory parallelism, OpenCL for heterogeneous devices, direct GPU programming using CUDA, and directive-based programming using OpenACC make up a complete toolset for parallel computing in the current age. When combined with strong compilation platforms like NVCC, those tools can help build highly-parallel, highly scalable applications that can take full advantage of CPUs and GPUs in clusters and HPC arrangements as well as smart modules/devices. Its flexibility in handling workload scheduling, memory, and execution in one or multiple computing architectures becomes important for AI, ML, and real-time analytical workloads.

4 Results and Discussion

In this section, we analyzed the results of PakSupercomputer (shown in Figure 5), and compared them with the existing system available in Pakistan. This particular section is additionally categorized into three subsections; the *System Performance*, *System Efficacy*, and the *System Ranking*.

4.0.1 System Performance

In this experiment, we executed three important benchmarks, the N-body, Reverse Time Migration [27], and Monte Carlo, on the proposed PakSupercomputer and measured its performance. While running these benchmarks on a PakSupercomputer with 80 Workers nodes, the results show that the system achieves 1.0291 Peta single precision Floating Point Operations per Second (FLOPS) and consumes dynamic power up to 25.35 Kilowatts.

4.0.2 Efficacy

In this section, we measure the efficacy of the proposed PakSupercomputer with commercial supercomputing solutions from established brands like Lenovo and Intel. To measure the efficiency of PakSupercomptuer Six parameters were considered; Programmability, Portability, Scalability, Performance, Power, and Cost. For better efficacy, the Programmability, Portability, Scalability, and Performance need to be higher and Power and Cost need to be lower shown in Figure 4. Programmability

Table 2: Available Super-Computer System Architectures in Pakistan

Institute	Peak Performance	Power Consumption	Technologies Used	Manufacturer
Namal, PakSupercomputer	1.1 PFLOPS	25 KW	Intel Xeon, GPU acceleration, Fast Eathernet	Custom-build
NUST, Islamabad [24]	650 TFLOPS	N/A	Intel Xeon, NVIDIA GPUs, Infiniband	HPE
UCERD, Islamabad	113 TFLOPS	N/A	AMD EPYC, NVIDIA GPUs, Fast Ethernet	Dell
PAK-IAST Haripure [25]	91 TFLOPS	N/A	Intel Xeon, NVIDIA GPUs, InfiniBand	Lenovo
KUST, Kohat	0.416 TFLOPS	N/A	Intel Xeon, basic networking	Custom-built
COMSATS, Islamabad	0.158 TFLOPS	N/A	Intel Cores, NVIDIA GPUs, standard network	HP
CIIT, Islamabad	0.05 TFLOPS	N/A	Intel processors	Custom-built
UoM, Malakand	NA	N/A	Intel processors	Custom-built
GIK Institute [26]	N/A	N/A	N/A	N/A
KRL	N/A	N/A	N/A	N/A
UET Lahore	N/A	N/A	N/A	N/A
NED Karachi	N/A	N/A	N/A	N/A

represents the efforts used while coding an application. Figure 4 shows that the proposed PakSupercomputer has higher efficacy than the branded supercomputer such as Lenovo, Dell Technologies. The results show that due to the efficient Elastic Programming Model, PakSupercomputer provides better programming support. Application portability is measured by deploying heterogeneous applications allowing the same program or software in different computer architectures. Due to an efficient distributed operating stack and application libraries, the PakSupercomptuer gives better portability. The Scalability validates the stable system performance with the increase in the number of cores, whereas the performance represents the speed-up of a system against the other. The results show that branded commercial supercomputer gives better performance as well as scalability due to the advanced and latest technologies used. The PakSupercomputer achieves good performance due to use of commercially off-the-shelf (COTS) having standard cost-effective hardware which are readily available and programming models. The Elastic Parallel Programming model not only improves the programmability but also increases the portability. The scalability is increased by using the OpenSource Software Stack and advanced hardware architecture which has the capability to handle and schedule thousands of cores.

The Power shown in the Figure 4 is consumed power while executing the applications and cost is the total cost of a supercomputer. The results confirm that the proposed supercomputer not only utilizes low power but also costs 3 times less than the commercial supercomputer solutions. The reason for this the Supercomputer uses COTS hardware, open-source distributed software and operating system stacks. The COTS hardware allow greater flexibility and faster adoption of the latest technologies in low price. The improvement in Performance with low energy and cost budget is achieved by using TPU and FPGA accelerators on heterogeneous nodes.

4.0.3 System Ranking

In Pakistan, different universities have established their supercomputer architectures. Some of the leading universities are NUST Islamabad, PAF-ISAT, GIK Instiute, COMSATS, CIIT Islamabad, UET Lahore, and NED Karachi. If we compare those universities with our proposed architecture with Peak Performance and Power Consumption, then we get the summary as shown in Table 2.

While comparing with the available supercomputing systems in Pakistan, the proposed PakSupercomputer delivers up to 1.1 PetaFLOPS with a power consumption of 25.2 KW, ranking it as the most powerful in the country. While NUST's supercomputer follows closely at around 650 TeraFLOPS, and UCERD reaches 113 TFLOPS. However, while comparing similar architectures globally [3], it has been estimated that the PakSupercomptuer outperforms most other supercomputers in the region in terms of both computational power and energy efficiency. This establishes and proves that PakSupercomputer is a key resource for advanced computing in Pakistan.

5 Conclusion

In this project, we have presented a new, energy-efficient, and affordable supercomputer system titled Scalable Heterogeneous Supercomputer (PakSupercomputer). The PakSupercomputer utilizes a mix of CPU, GPU, and FPGA processing units to achieve optimal performance without sacrificing energy efficiency. In order to guarantee flexibility and efficiency in utilizing diverse hardware, we have created a software stack that accommodates OpenMP, OpenACC, and OpenCL programming models. Our comparison with current supercomputers in Pakistan shows positive outcomes. In terms of performance, the developed supercomputing system architecture stands at second place out of all the systems that were reviewed while comparing performance per watt, making it the top performer. These results show that our proposed PakSupercmputer provides a feasible option for developing advanced computing capabilities without facing expensive costs or high energy requirements. The PakSupercomputer is a major advancement in bringing supercomputing technology to under-developed regions, allowing for advanced computational research and applications.

6 Acknowledgements

This work was supported by the Higher Education Commission - Technology Development Fund (HEC-TDF), Unal Color of Education Research and Development (UCERD) Private Limited, Islamabad, and the Namal Education Foundation. The authors gratefully acknowledge the Centre for AI and Big Data at Namal University and the Barcelona Supercomputing Centre for providing technical support and computational resources for this research.

7 References

[1] Moore, Gordon E and others, "Cramming more components onto integrated circuits," 1965.

[2] Md. Amin and Al. Tanvir, "Multi-core: Adding a New Dimension to Computing," *arXiv preprint arXiv:1011.3382*, 2010.

[3] Top500 Organization, "Top500 Supercomputer Sites." `https://www.top500.org`, 2024. Accessed: 2024-10-12.

[4] S. Craven and P. Athanas, "Examining the viability of fpga supercomputing," *EURASIP Journal on Embedded systems*, vol. 2007, no. 1, pp. 13–13, 2007.

[5] X. Tian and K. Benkrid, "Massively parallelized quasi-monte carlo financial simulation on a fpga supercomputer," in *High-performance reconfigurable computing technology and applications, 2008. HPRCTA 2008. Second International Workshop on*, pp. 1–8, IEEE, 2008.

[6] N. Rajovic, P. M. Carpenter, I. Gelado, N. Puzovic, A. Ramirez, and M. Valero, "Supercomputing with commodity cpus: Are mobile socs ready for hpc?," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, p. 40, ACM, 2013.

[7] N. Rajovic, A. Rico, N. Puzovic, C. Adeniyi-Jones, and A. Ramirez, "Tibidabo: Making the case for an arm-based hpc system," *Future Generation Computer Systems*, vol. 36, pp. 322–334, 2014.

[8] A. O Mahony, B. Hanzon, and E. Popovici, "The role of fpgas in modern option pricing techniques: A survey," *Electronics*, vol. 13, no. 16, p. 3186, 2024.

[9] Y. Kondo, H. Sato, and S. Matsuoka, "Large-scale distributed sorting for gpu-based heterogeneous supercomputers," in *2014 IEEE International Conference on Big Data (Big Data)*, pp. 116–125, IEEE, 2014.

[10] D. Milojicic, P. Faraboschi, N. Dube, and D. Roweth, "Future of hpc: Diversifying heterogeneity," in *DATE 2019*, 2019.

[11] M. Riedel, R. Sedona, C. Barakat, P. Einarsson, R. Hassanian, G. Cavallaro, M. Book, H. Neukirchen, and A. Lintermann, "Practice and experience in using parallel and scalable machine learning with heterogenous modular supercomputing architectures," in *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2021.

[12] R. M. Chang, R. J. Kauffman, and Y. Kwon, "Understanding the paradigm shift to computational social science in the presence of big data," *Decision Support Systems*, vol. 63, pp. 67–80, 2014.

[13] T. Hussain, M. Pericas, and E. Ayguadé, "Reconfigurable memory controller with programmable pattern support," in *5th HiPEAC Workshop on Reconfigurable Computing (WRC), Heraklion Crete 2011*, vol. 67, p. 95, 2011.

[14] T. Hussain, "Hmmc: A memory controller for heterogeneous multi-core system," *Microprocessors and Microsystems*, vol. 39, no. 8, pp. 752–766, 2015.

[15] T. Hussain, O. Palomar, A. Cristal, E. Ayguade, and A. Haider, "Access pattern based multi-layer bus virtualization controller," in *The 13th International Bhurban Conference on Applied Sciences & Technology*, 2016.

[16] T. Hussain, "A novel hardware support for heterogeneous multi-core memory system," *Journal of Parallel and Distributed Computing*, vol. 106, pp. 31–49, 2017.

[17] T. Hussain, O. Palomar, A. Cristal, E. Ayguade, and A. Haider, "Mvpa: An fpga based multi vector processor architecture," in *The 13th International Bhurban Conference on Applied Sciences & Technology*, IEEE, 2016.

[18] T. Hussain, "Ppmc: On chip memory manager and scheduler for vector processor," *Appeared in HiPEAC info (issue 32): October 2012.*, 2012.

[19] T. Hussain, A. Haider, and E. Ayguade, "Pmss: A programmable memory system and scheduler for complex memory patterns," *Journal of Parallel and Distributed Computing*, 2014.

[20] T. Hussain, O. Palomar, A. Cristal, O. S. Unsal, A. Eduard, and V. Mateo, "Pams: Pattern aware memory system for embedded systems," in *2014 International Conference on Reconfigurable Computing and FPGAs (ReConFig 2014)*, IEEE, 2014.

[21] T. Hussain, A. Haider, S. Gursal, A., and E. Ayguade, "Amc: Advanced multi-accelerator controller," *Publisher : Journal of Parallel Computing*, 2014.

[22] W. Akram, T. Hussain, and E. Ayguade, "Fpga and arm processor based supercomputing," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pp. 1–5, IEEE, 2018.

[23] H. Tassadaq, T. Muhammad, Wasay, Q. Abdul, and A. Eduard, "Design and development of risc-v based virtual cluster using qemu simulator," in *21st International Conference on Frontiers of Information Technology (FIT'24)*, p. 8, 2024.

[24] N. U. of Sciences & Technology (NUST), "Supercomputing research & education center (screc), research center for modeling & simulation (rcms)," 2024. Performance: 650 TeraFLOPS, Accessed: 2024 November.

[25] S.-P. C. for Artificial Intelligence, "Sino-pak center for artificial intelligence (spcai)," 2024. Cost: 190 Million PKR, Performance: 91 TeraFLOPS.

[26] G. Institute, "High performance computing cluster," 2024. Accessed: 2024-11-09.

[27] T. Hussain, M. Pericas, N. Navarro, and E. Ayguadé, "Implementation of a reverse time migration kernel using the hce high level synthesis tool," in *Field-Programmable Technology (FPT), 2011 International Conference on*, pp. 1–8, IEEE, 2011.