University of Essex

Department of Computer Science and Electronic Engineering

Project Proposal Submitted as part of

CE902 Professional Practise and Research Methodology

A Hardware and Software Task-Scheduling Framework Based on CPU+FPGA+GPU Heterogeneous Architecture in Edge

Submitted on March 22, 2024

Supervisor

Dr Sangeet Saha

Assessor

Dr Roneel Sharan

Author

Harshini Bangalore Amarnath

(Registration **2311849**)

ABSTRACT

In today's data driven era, processing vast amounts of data efficiently has become increasingly crucial. Traditional computing systems, primarily reliant on CPUs, struggle to meet the escalating demands for processing power. Edge computing has emerged as a strategic response to the limitations of centralized data processing. By executing computations closer to the data source, edge computing minimizes latency and bandwidth consumption, making it ideal for real-time applications across various sectors such as autonomous vehicles, industrial automation, healthcare, and smart cities. This project focuses on developing a comprehensive task-scheduling framework tailored for edge computing environments. Leveraging the unique strengths of CPUs, GPUs and FPGAs, the framework aims to optimize performance and resource utilization. Tasks are intelligently allocated to the most suitable processing element using Jetson Nano. The development of this task scheduling framework aims to significantly contribute to the advancement of edge computing technology. By addressing the increasing demand for efficient resource utilization and performance optimization in heterogenous computing environments, the framework paves the way for the development of faster, more responsive edge computing applications, composed to meet the evolving needs of modern pattern.

Keywords

CPU, GPU, FPGA, edge computing, task scheduling, heterogeneous system, framework, Jetson Nano.

TABLE OF CONTENTS

1 INTRODUCTION

In today's rapidly evolving technological landscape, the demand for efficient and high-performance computing systems has never been greater. With the expansion of data intensive applications and the emergence of edge computing, there is a growing need for systems capable of processing vast amounts of data quickly and reliably. Edge computing systems, characterized by their ability to perform computation closer to the data source, have become essential in various domains, ranging from autonomous vehicles and industrial automation to healthcare and smart cities. Traditionally, computing systems relied heavily on Central Processing Units (CPUs) to perform most tasks. However, the exponential growth in data volume and complexity has required for the integration of specialized accelerators to argument CPU capabilities. These accelerators, including Graphics Processing Units (GPUs), Application Specific Integrated Circuit (ASIC) and Field Programmable Gate Array (FPGAs), are instrumental in enhancing computational efficiency and accelerating specific workloads.

Among these accelerators, GPUs have gained prominence for their ability to handle parallel tasks efficiently. With thousands of cores optimized for parallel processing, GPUs excel at tasks such as image and video processing, machine learning, and scientific simulations. By offloading parallelizable tasks from the CPU to the GPU, overall system performance can be significantly enhanced, enabling faster execution of complex computations. Similarly, FPGAs offer unique advantages in terms of flexibility and energy efficiency. Unlike fixed-function ASICs, FPGAs can be reconfigured spontaneously to adapt to different computational tasks. This inherent flexibility allows FPGAs to dynamically allocate resources based on workload requirements, resulting in optimized performance and reduced energy consumption. Additionally, FPGAs are well-suited for implementing custom logic and accelerating specific algorithms, making them ideal for edge computing applications with diverse processing demands. To validate the effectiveness of such systems Jetson online Emulator through PyPI, offers a virtual environment to simulate the functionality of NVIDIA's Jetson Nano embedded computing platform.

By developing task-scheduling framework, the aim is to contribute to the advancement of edge computing technology. This framework seeks to address the growing need for the efficient resource utilization and performance optimization in heterogeneous computing environments, ultimately facilitating the development of faster, more responsive edge computing systems.

2 RELATED WORKS

In recent years, there has been a growing interest in developing software solutions for edge computing environments, Specifically, there is a focus on creating task-scheduling frameworks that leverage heterogeneous architectures combing CPUs, FPGAs and GPUs. There are many research papers that has been published to contribute these issues. In this section, a review will be made regarding task scheduling, CPU+GPU+FPGA architectures, frame works and edge computing.

A paper based on improving how tasks are scheduled between regular CPUs, GPUs and FPGAs in edge computing system was published in 2019 [1]. They created a framework called Dynamic Partial Reconfiguration (DPR) to better manage these tasks, considering factors like how long it takes to switch between tasks and how predictable the FPGA tasks are. They tested this framework on a specific platform called Zynq and found that it greatly Improved efficiency. But there could have included more ways of using FPGAs alongside regular processor, like as part of the main processor or as an additional processor. Additionally in their project, they also could have integrated on other types of processors like GPUs.

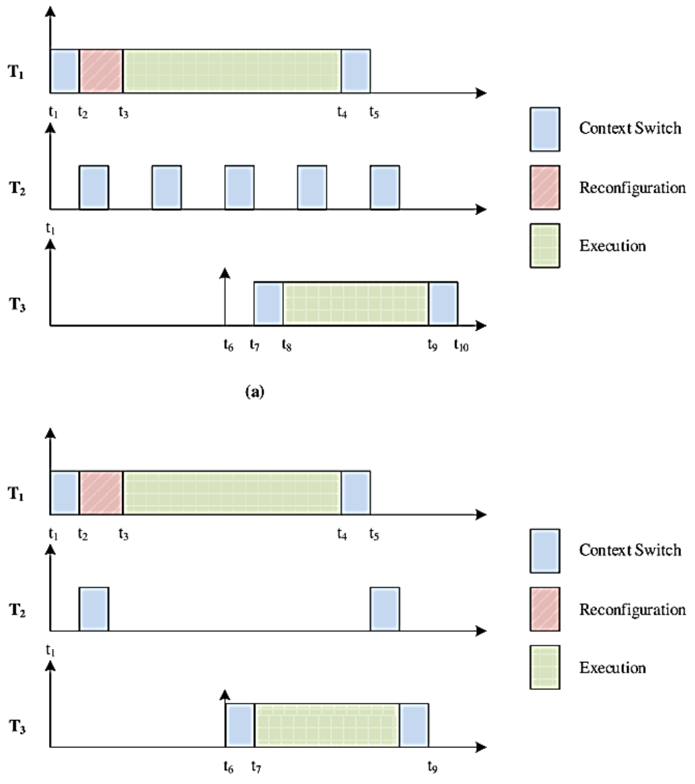


Figure 1: *Task graph allocation* [1]

In 2020 there was research another research which was focused about improving performance and energy efficiency by scheduling tasks on CPU and FPGA processors [2]. They developed a scheduling framework in C++ to efficiently manage tasks on these heterogeneous platforms, achieving significant performance improvements compared to using only CPUs or FPGAs individually. They also integrated a novel algorithm the n-body problem using both CPU and FPGA resources. Also, enhancement to optimize energy efficiency dynamically and extending it to target platforms combining CPUs, GPUs and FPGAs could have been done. It's important to note that, even here they did not specifically research GPUs in this project.

Later in the same year 2020 there were two research which were done GPU. First one was the research which conducted experiments comparing the performance of applications of NVIDIA's Jetson TX2 embedded GPU with a high-end-desktop GPU, the Titan X [3]. They tested various applications from different domains to understand Jetson's capabilities. Despite the Jetson TX2 having fewer cores, memory, and speed compared to the TITAN X, they found that it could achieve comparable performance for certain applications. This study highlights Jetson's potential beyond limited resource scenarios. However, they could have provided more detailed insights into the specific types of applications where Jetson excels, and perhaps explored optimizations to improve its performance. Secondly, [4] investigated about the energy-efficient scheduling techniques for heterogeneous computing systems combining CPUs and GPUs. They developed a model to understand CPU-GPU utilization dynamics and proposed a heuristic greedy strategy (UEJS) and a hybrid particle swarm optimization algorithm (H-PSO) to reduce energy consumption during job scheduling. However, they could have provided clearer explanation of the architecture and models used. Additionally, they successfully compared their proposed algorithms with existing ones but could have included more details on how task mapping was performed to further validate their results.

Further in 2021 the researchers addressed the challenge of mapping multiple applications onto CPUs and GPUs in embedded systems efficiently. In [5] we can see that this project proposed a fine-grain mapping framework and practical algorithms to minimize completion time. By evaluating their approach with various benchmarks, including real-world neural networks, they demonstrated significant speed improvements compared to existing techniques. However, they could have provided more detailed insights into the specific factors expected in their mapping framework and explained how their algorithms achieve faster completion times in different scenarios. Also, a clearer explanation of the experimental setups, benchmarks used, and performance comparison of CPU-GPU would have enhanced the understanding of their results.

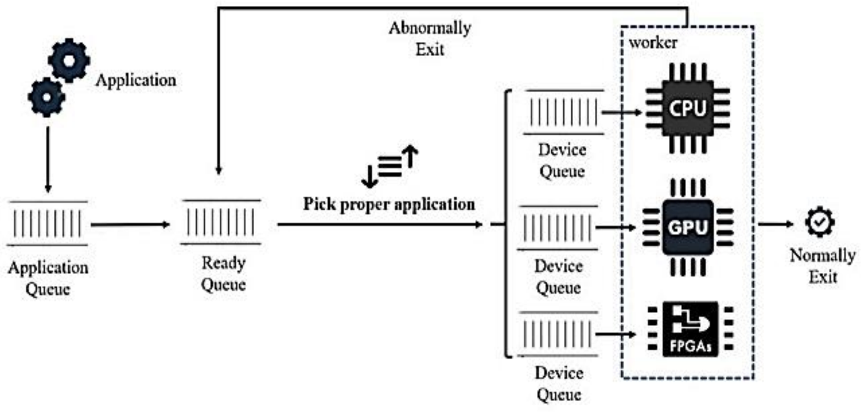


Figure 2: *General structure of heterogeneous schedular* [5]

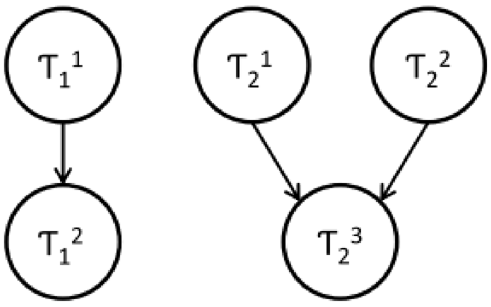


Figure 3: *Task dependency graph* [5]

In 2023, performance comparison of CPU and GPGPU calculations was performed using three simple case studies for various parallel programming tasks [6]. The results showed that for certain tasks like matrix multiplication, CPUs were more efficient for smaller matrices, while for larger matrices GPUs performance was good. However, for image processing tasks, CPUs with SMID instructions outperformed GPUs. They suggested recommendations based on their findings but could have included a more extensive range of algorithm and datasheets for a comprehensive comparison. Additionally, exploring the impact of different operating systems and compilers could have provided deeper insights into performance variations. Also, task scheduling technique for energy-constrained edge devices could be explored.

Now in 2024, [7] multiobjective energy-efficient task scheduling technique (METSEM) task scheduling technique for edge devices aiming to reduce energy and time overhead was proposed. They could have included a comparison with CPU and GPU implementations to explore potential performance improvement.

A research on CPU and GPU parallel efficiency was analysed in 2024 [8] for a cluster comprising 16 Raspberry Pi 4 boards and 8 Nvidia Jetson Nano Boards, evaluating both CPU and GPU implementation for Computational Fluid Dynamics (CFD) applications. They found that the parallelization scales effectively with the number of boards, achieving high efficiency. However, the description does not provide specific details about how this heterogeneous architecture was utilized or optimized for task scheduling and parallel efficiency, which could have been beneficial for a comprehensive understanding of the study’s findings.

These studies provide valuable insights in optimizing task scheduling in heterogeneous computing environments, focusing on combinations of CPUs, GPUs and FPGAs. By understanding their approaches and findings, this project designs a comprehensive task-scheduling framework, leveraging the capabilities of a Jetson Nano embedded GPU. The gap that project is filling is integrating and optimizing task scheduling framework across CPU, GPU and FPGA tailored to edge devices like Jetson Nano.

3 BACKGROUND

3.1 PROJECT CONTEXT

Edge Computing: Edge computing involves processing data closer to its source rather than relying on centralized data centres. It aims to reduce latency and bandwidth usage by performing computations near the data generation point. In today's data-driven world, edge computing is critical for applications requiring real-time processing and low-latency responses. It is particularly essential in domains like autonomous vehicles, industrial automation, healthcare and smart cities.

Evolution of computer architectures:

- Traditional CPU-Centric Systems: Historically, computing systems relied heavily on CPUs to execute tasks. However, the increasing complexity and volume of data have necessitated the integration of specialized accelerators.
- Emergence of Specialized Accelerators (GPU, FPGA, ASIC): These have emerged as accelerators to enhance computational efficiency and performance.
- Need of Heterogeneous Integration: Heterogeneous Integration involves combining CPUs, GPUs and FPGAs to leveraging their respective strengths for optimized performance and energy efficiency in edge computing scenarios.

Role of GPUs in Edge Computing: GPUs are highly efficient at parallel processing, making them ideal for tasks such as image processing, machine learning and scientific simulations. GPUs excel in accelerating compute-intensive tasks, including deep learning interference, speech recognition and large-scale simulations by offloading tasks from CPU. By offloading parallelizable tasks to GPUs, overall system performance overall system performance can be significantly improved, enabling faster execution of complex computations, and reducing the burden on CPUs.

Advantages of FPGAs in Edge Computing: FPGAs are known for their energy efficiency, making them suitable for edge computing environments where power consumption is a concern. FPGAs enable the implementation of custom logic tailored to specific edge computing applications, allowing for optimized performance and resource utilization.

Task Scheduling Framework: This is to enhance edge computing performance through efficient task allocation and management. The framework aims to optimize resource utilization and maximize computational efficiency in edge computing environments. Framework components are as follows:

- Software task allocation on CPUs: The framework will intelligently allocate software tasks to CPUs based on their characteristics and requirements. Tasks that are sequential or not suitable for acceleration will be assigned to the CPU for execution.
- Hardware Task Offloading to GPUs and FPGAs: Hardware-accelerated tasks will be offloaded to GPUs and FPGAs to leverage their parallel processing capabilities and energy efficiency. These accelerators will handle intensive tasks that can benefit from parallel execution.
- Dynamic Reconfiguration with DPR: DPR will be utilized to enable selective reconfiguration of FPGA regions while the system is operational. This dynamic reconfiguration allows for seamless adaption to changing computational demands, optimizing performance and energy efficiency.

3.2 TECHNICAL CONTEXT

In edge computing projects, the Jetson Nano and Jetson Emulator offer valuable resources for developing and testing AI applications. Their integration allows developers to explore AI edge computing without investing in physical hardware initially.

Jetson Emulator [9]: Jetson emulator serves as a conventional tool for simulating the functionality of NVIDIA’s Jetson AI-Computer’s Interference and Utilities API. It provides a pre-configured environment interference task, allowing developers to test applications and algorithms in a simulated setting. It also serves as a virtual environment for image classifications, object detection and image segmentation tasks. This emulator requires no setup and operates offline, eliminating the need for configuring firewall ports.

Jetson Nano [10]: Jetson Nano provides the computational power necessary for running neural networks and processing AI tasks efficiently. Remarkably, it operates on as little as 5watts of power, making it energy efficient. With its low power consumption and hight performance, it is well suited for edge computing scenarios where energy efficiency is crucial.

4 PROBLEM SATEMENT

Given the set of tasks like matrix multiplication, sorting and three processing elements (CPU, GPU, and FPGA), my objective is to develop an intelligent mapping or allocation scheme so as to get efficient allocation of task across the processing elements.

4.1 RESEARCH QUESTIONS

1. Finding the characteristic features of CPUs, GPUs in Jetson emulator as well as Jetson Nano and FPGA characteristics on Zynq platform.
2. Key factors influencing task allocation decisions in heterogeneous edge computing environments so that task scheduler effectively utilize machine learning models to dynamically allocate tasks to CPUs, GPUs and FPGAs based on their individual characteristics.
3. What are the performance implications of the proposed task scheduling framework in terms of resource utilization, latency reduction and overall system throughput?
4. How does the proposed framework compare to existing approaches in terms of computational efficiency, energy consumption and scalability?

5 GOALS

5.1 OBJECTIVES

The main aim of this master's dissertation project is to develop a comprehensive task-scheduling framework leveraging CPU, FPGA and GPU resources within edge computing environments. The framework targets efficient hardware and software integration to optimize performance and resource utilization in heterogeneous architecture.

To achieve this the project will go through several steps that include:

1. Determining the characteristic features on CPUs, GPUs and FPGAs like memory consumption, out-of-order (ooo) execution cycles and instructions processed.
2. Employ machine learning techniques to analyse and model the performance characteristics of CPUs, GPUs and FPGAs using collected data.
3. Determine key factors influencing task allocation decisions in heterogeneous edge computing environments, considering factors like workload type, resource availability and latency requirements.

5.2 SCOPE AND LIMITATIONS

This dissertation project aims to develop a comprehensive task-scheduling framework based on CPU, FPGA and GPU heterogeneous architecture in edge computing environments. The scope of this project includes : Characterization of hardware components, machine learning analysis, task allocation decisions, development of task-scheduling framework and evaluation.

However, the project has limitations such as the experiments will be conducted on Jetson emulator and Jetson Nano platforms, which may not-fully represent the real-world scenarios, the experiments will focus on specific tasks like matrix multiplication, potentially limiting the generalizability of the findings to other applications. If the data set size is not minimum, it is very difficult to extract exact features of processing elements which affects scheduling. The proposed task-scheduling framework may not fully capture the complexities of real-world edge computing scenarios, leading to potential oversimplification. The project may face limitations in terms of available resources, time constraints and access to specialized hardware for applications.

6 METHADOLOGY

6.1 PROJECT FLOW METHADOLOGY

1. Collecting characteristic features of CPU, GPU, and FPGA

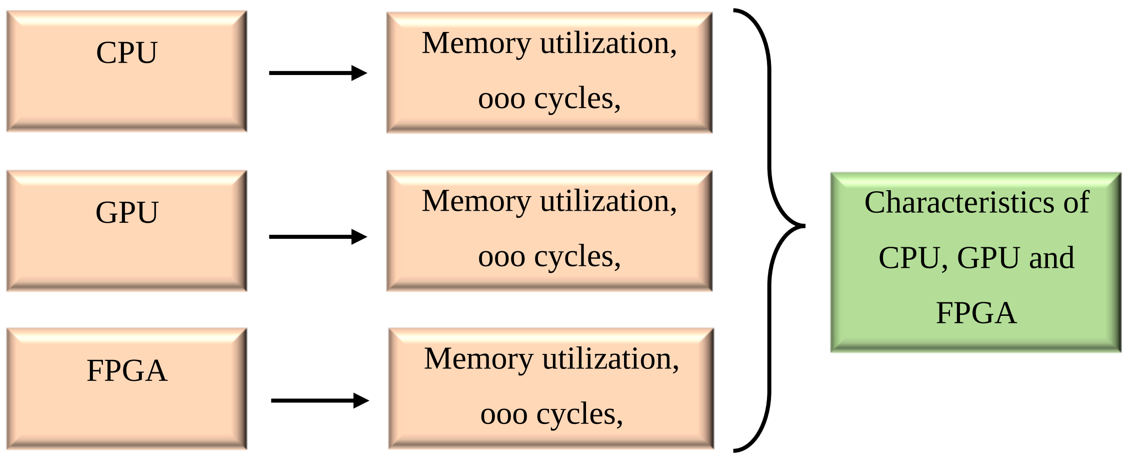
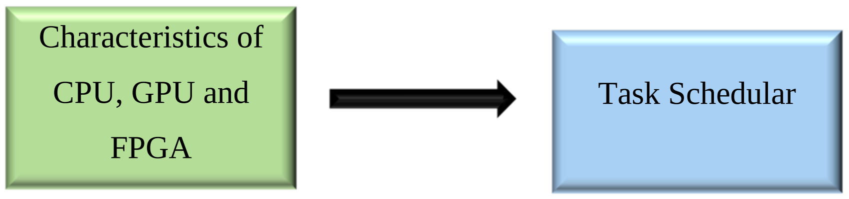

CPU
Memory utilization, ooo cycles,
GPU
Memory utilization, ooo cycles,
FPGA
Memory utilization, ooo cycles,
Characteristics of CPU, GPU and FPGA

Figure 4: *Characteristics of CPU, GPU and FPGA*

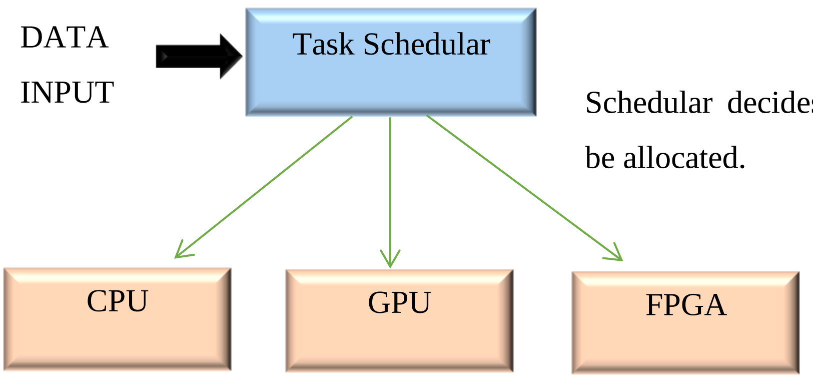
2. Training task schedular to analyse the performance of CPU, GPU, and FPGA



Using Machine learning, task schedular decides which task to be allocated either to CPU,

Figure 5: *Task schedular analysing performance of CPU, GPU and FPGA using ML*

3. Once the task schedular is trained, whenever the input is given task schedular will be allocating tasks either to GPU, CPU or FPGA.



Schedular decides which task should be allocated.

Figure 6: *Methodology flow*

6.2 PROGRESS UPDATE

Till date, the following tasks are done.

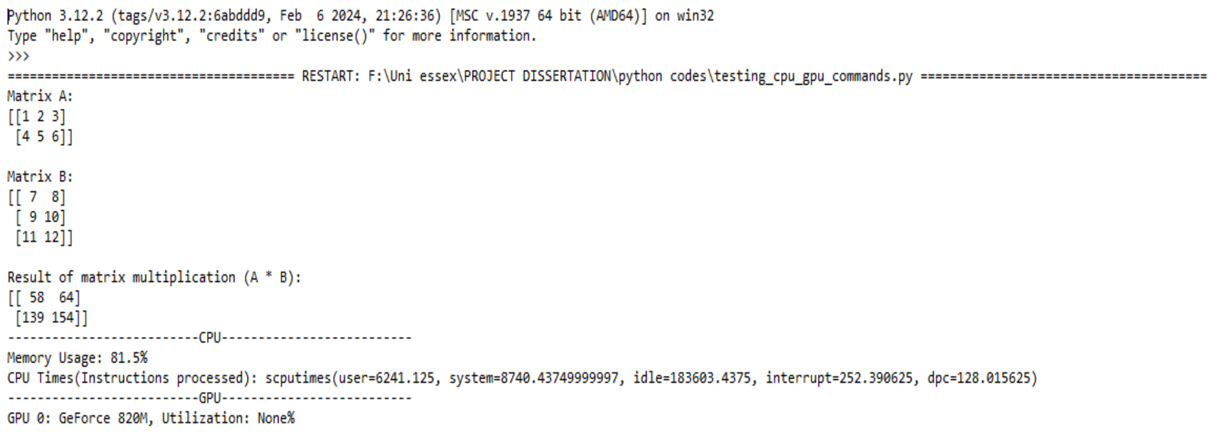
CPU memory usage, number of instructions processed and GPU memory consumption for matrix multiplication in Jetson Online emulator was tested using Jetson libraries. As expected ooo cycles and other characteristics of GPU should be run on Ubuntu Linux platform.

Jetson libraries:

import setuptools

import jetson_emulator.inference as inference

import jetson_emulator.utils as utils



```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======================================= RESTART: F:\Uni essex\PROJECT DISSERTATION\python codes\testing_cpu_gpu_commands.py =======================================
Matrix A:
[[1 2 3]
 [4 5 6]]

Matrix B:
[[ 7  8]
 [ 9 10]
 [11 12]]

Result of matrix multiplication (A * B):
[[ 58  64]
 [139 154]]
-------------------------CPU-------------------------
Memory Usage: 81.5%
CPU Times(Instructions processed): scputimes(user=6241.125, system=8740.43749999997, idle=183603.4375, interrupt=252.390625, dpc=128.015625)
-------------------------GPU-------------------------
GPU 0: GeForce 820M, Utilization: None%
```

Figure 7: *Output of the calculated matrix multiplication on Jetson emultor*

Here is an example table in which the output characterisctics of CPU and GPU will be updated in the below table:

MATRIX MULTIPLICATION	**Memory usage**	**Ooo cycles**	**Instructions Processed**	**Disk usage**
CPU				
GPU				

Table 1: *CPU and GPU output characteristics*

7 EVALUATIONS

This section will discuss the evaluation methods that will be used to ensure that the proposed method has been built successfully. The performed evaluation process will help in understanding the achieved objectives and goals of the project. The main objectives of that evaluation task are to make sure that each part of the project is working perfectly and to get the desired performance.

7.1 UNIT TESTING

Unit test will be conducted using some of the benchmarks like matrix multiplication, sorting, hashing on individual processing elements i.e., CPU, GPU, and FPGA.

7.2 INTEGRATION TESTING

Once unit testing is completed, each of the characteristics of individual processing elements are analysed and fed to task schedular. This process is facilitated by machine learning algorithms so that the task schedular can intelligently allocate tasks to the respective processing elements based on their capabilities.

7.3 VALIDATION TEST

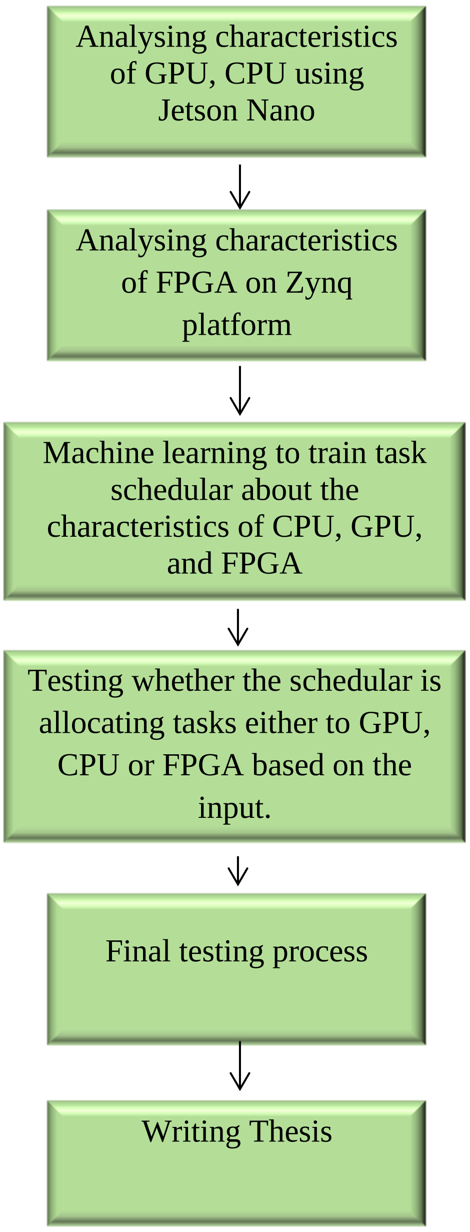
Validation testing is performed on EEMBC (Embedded Microprocessor Benchmark Consortium) [11]. These benchmarks help standards and performs reliably under various workloads.

8 PROJECT PLAN

Since project timeline is a very essential part of any proposal, we will be tracking our project development process and manage all the objective through two methods:

Work breakdown structure and a Gantt Chart which will be shown in this section:

8.1 WORK BREAKDOWN STRUCTURE



Analysing characteristics of GPU, CPU using Jetson Nano
Analysing characteristics of FPGA on Zynq platform
Machine learning to train task schedular about the characteristics of CPU, GPU, and FPGA
Testing whether the schedular is allocating tasks either to GPU, CPU or FPGA based on the input.
Final testing process
Writing Thesis

Figure 8: *Work Breakdown Structure*

8.2 GANTT CHART

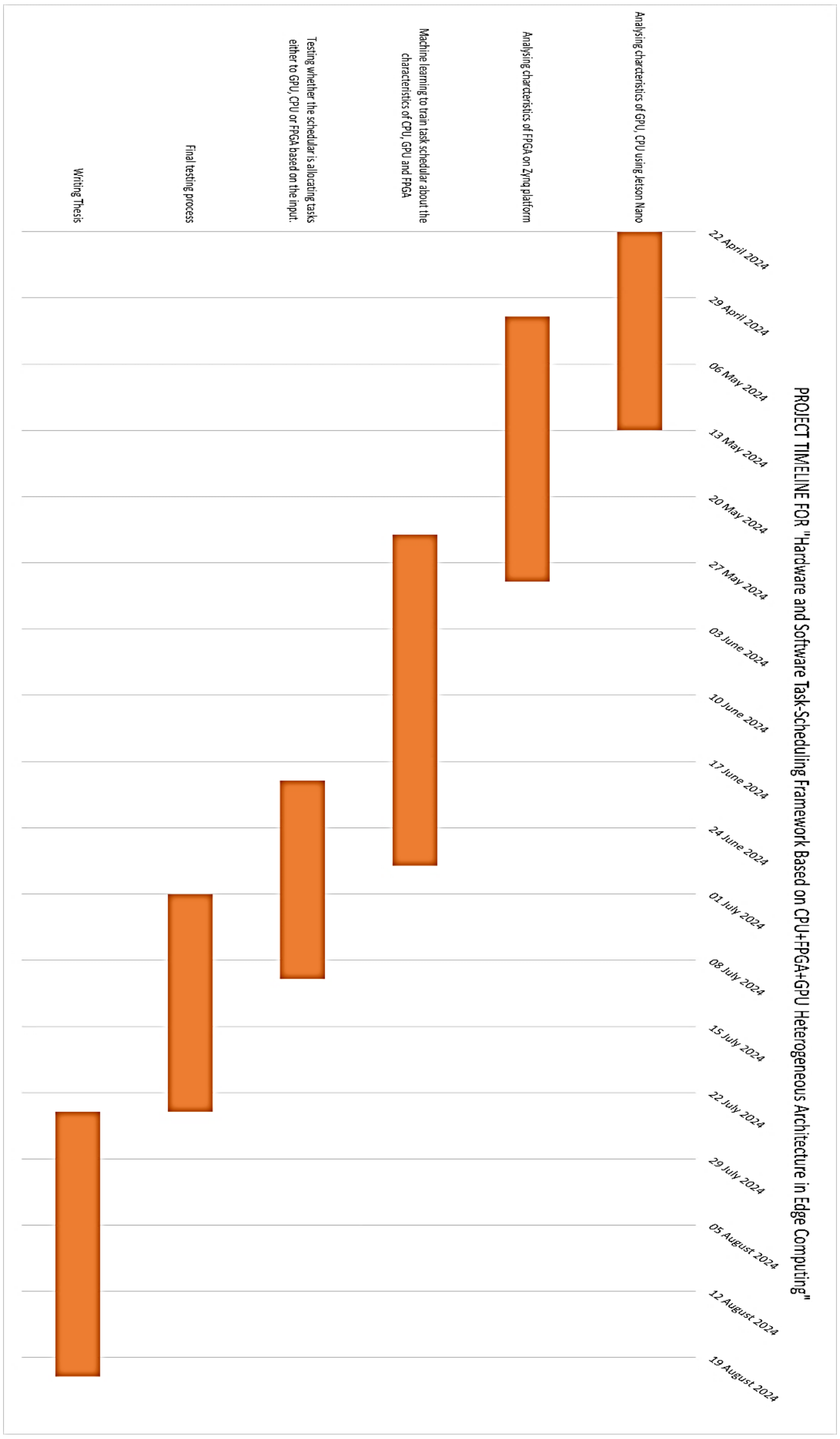


Figure 9: *Gantt Chart for the Project Timeline*

9 CONCLUSION

In conclusion, this project proposal outlines the development of a comprehensive task scheduling framework tailored for edge computing environments. With the increasing demand for efficient resource utilization and performance optimization in heterogeneous computing systems, the proposed framework aims to address these challenges by intelligently allocating tasks to CPUs, GPUs and FPGAs based on their individual characteristics.

Although there are existing studies on task scheduling in heterogeneous computing environment, there is gap in research specifically focused on edge computing platforms like Jetson Nano, Jetson Emulator with CPU+GPU and FPGA framework. This project aims to fill that gap by developing a comprehensive framework tailored for such platforms, ultimately contributing to the advancement of edge computing technology.

In future work, the focus will be on refining the task scheduling framework to enhance its efficiency and adaptability across various edge computing scenarios. Integrating the framework with edge AI applications, such as those in autonomous vehicles and healthcare will showcase its ability to optimize performance and resource allocation.

In summary, this project facilitates development of faster, more responsive edge computing applications. Though diligent research, implementation and evaluation, the proposed task scheduling framework has the potential to make meaningful contributions to the field of edge computing.

10 REFERENCES

[1] Zongwei Zhu, Junneng Zhang, Jinjin Zhao, Jing Cao, Duan Zhao, Gangyong Jia (Member, Ieee), And Qingyong Meng, “A Hardware and Software Task-Scheduling Framework Based on CPU+FPGA Heterogeneous Architecture in Edge Computing,” 2019.

[2] Andrés Rodríguez, Angeles Navarro1, Rafael Asenjol, Francisco Corbera1, Rubén Gran, Darío Suárez2, Jose Nunez-Yanez3, “Parallel multiprocessing and scheduling on the heterogeneous Xeon+FPGA platform,” 2020.

[3] A. Ozsoy, “A Comprehensive Performance Comparison of Dedicated and Embedded GPU Systems,” 2020.

[4] Xiaoyong Tang And Zhuojun Fu, “CPU–GPU Utilization Aware Energy-Efficient Scheduling Algorithm on Heterogeneous Computing Systems,” 2020.

[5] Zexin Li, Yuqun Zhang, Ao Ding, Husheng Zhou, Cong Liu “Efficient algorithms for task mapping on heterogeneous CPU/GPU platforms for fast completion time,” vol. 114, 2021.

[6] Branislav Lipovsky, Slavomir Simonak, “Performance comparison of CPU and GPGPU calculations using three simple case studies,” Vols. vol.31, no.1(91), 2023.

[7] Qiangqiang Jiang, Xu Xin, Libo Yao, Bo Chen, “METSM: Multiobjective energy-efficient task scheduling model for an edge heterogeneous multiprocessor system,” 2024.

[8] Bastien Di Pierro, Sarah Hank “CPU and GPU parallel efficiency of ARM based single board computing cluster for CFD applications,” 2024.

[9] https://pypi.org/project/jetson-emulator/

[10] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[11] https://www.eembc.org/