Computer Systems
Science and Engineering

Tech Science Press

Doi:10.32604/csse.2025.061118

ARTICLE

# An Enhanced Task Migration Technique Based on Convolutional Neural Network in Machine Learning Framework

**Hamayun Khan[1,*], Muhammad Atif Imtiaz[2], Hira Siddique[3], Muhammad Tausif Afzal Rana[4], Arshad Ali[5], Muhammad Zeeshan Baig[6], Saif ur Rehman[7] and Yazed Alsaawy[5]**

[1]Department of Computer Science, Faculty of Computer Science & IT, Superior University, Lahore, 54000, Pakistan
[2]School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, Wollongong, NSW 2522, Australia
[3]School of Mathematics and Applied Statistics, University of Wollongong, Wollongong, NSW 2522, Australia
[4]School of Information Technology, King's Own Institute, Sydney, NSW 2000, Australia
[5]Faculty of Computer and Information Systems, Islamic University of Madinah, Al Madinah Al Munawarah, 42351, Saudi Arabia
[6]Department of Information Technology, Wentworth Institute of Higher Education, Sydney, NSW 2000, Australia
[7]Faculty of Electrical Engineering and Technology, Superior University, Lahore, 54000, Pakistan

*Corresponding Author: Hamayun Khan. Email: hamayun.khan@superior.edu.pk

Received: 18 November 2024; Accepted: 17 January 2025; Published: 19 March 2025

**ABSTRACT:** The migration of tasks aided by machine learning (ML) predictions IN (DPM) is a system-level design technique that is used to reduce energy by enhancing the overall performance of the processor. In this paper, we address the issue of system-level higher task dissipation during the execution of parallel workloads with common deadlines by introducing a machine learning-based framework that includes task migration using energy-efficient earliest deadline first scheduling (EA-EDF). ML-based EA-EDF enhances the overall throughput and optimizes the energy to avoid delay and performance degradation in a multiprocessor system. The proposed system model allocates processors to the ready task set in such a way that their deadlines are guaranteed. A full task migration policy is also integrated to ensure proper task mapping that ensures inter-process linkage among the arrived tasks with the same deadlines. The execution of a task can halt on one CPU and reschedule the execution on a different processor to avoid delay and ensure meeting the deadline. Our approach shows promising potential for machine-learning-based schedulability analysis enables a comparison between different ML models and shows a promising reduction in energy as compared with other ML-aware task migration techniques for SoC like Multi-Layer Feed-Forward Neural Networks (MLFNN) based on convolutional neural network (CNN), Random Forest (RF) and Deep learning (DL) algorithm. The Simulations are conducted using super pipelined microarchitecture of advanced micro devices (AMD) XScale PXA270 using instruction and data cache per core 32 Kbyte I-cache and 32 Kbyte D-cache on various utilization factors $(u_i)$ 12%, 31% and 50%. The proposed approach consumes 5.3% less energy when almost half of the CPU is running and on a lower workload consumes 1.04% less energy. The proposed design accumulatively gives significant improvements by reducing the energy dissipation on three clock rates by 4.41%, on 624 MHz by 5.4% and 5.9% on applications operating on 416 and 312 MHz standard operating frequencies.

**KEYWORDS:** Convolutional neural network (CNN); energy conversation; dynamic thermal management; optimization methods; ANN; multiprocessor systems-on-chips; artificial neural networks; artificial intelligence; multi-layer feed-forward neural network (MLFNN); random forest (RF) and deep learning (DL)

## 1 Introduction

Energy-aware multiprocessor systems-on-chips (MPSoCs) are widely used for multimedia and gaming embedded systems. Multi-core processors are rapidly turning into the norm for embedded real-time systems because of the demand for multimedia and gaming systems. These processors take into consideration greater adaptability while making undertakings, giving a substantially more productive environment for task assignments [1]. Task dispatching is a significant part of multiprocessors this article presents a dividing technique for reducing the energy of CPU in multi-core frameworks that reduces the worst-case execution time (WCET) and the absolute energy utilization of the framework. Our calculation is contrasted with others and we exhibit its prevalence [2].

MPSoC consists of multiple processors and all the components on the same chip consume low energy and require less power because of the tightly integrated architecture [3]. The consumption of MPSoC occurs in many abstractions from the logic level to the circuit. It's an integrated circuit and is designed for application with specific special software and hardware [4]. Proposed a scheduling technique for execution and assessment of CPU energy and memory of the central framework that is associated with multimedia applications. The proposed scheduler is dependent on a constant bandwidth server (CBS) [5]. Introduce an energy-efficient CPU scheduler that maps data using the earliest deadline first-based window constraint migration (EDF-WM) for multiprocessors in which memory reservation instrument uses paging calculations called shared anonymous private pages (PSAP) [6]. Introduce an energy-aware technique for the smooth allocation of the task. Task allocation expands the addition recurrence (usefulness) of its appointed tasks, which brings about expanded CPU clock recurrence [7]. Introduced an efficient processor with chip-packaged-based multi-core processing units, in which at least two processors are utilized to complete execution in equal intervals. This advancement in CPUs gives fast response and quicker execution times [8]. Introduce a Tasks scheduling based on logical link control (LLC) using Convolutional Neural Network (CNN) as shown in Fig. 1 based on the *m* processor scheduling algorithm (MPSA), L1-D, LLC and NOC Route showing multi-task mapping that creates an ideal schedule that expands the throughput of programs while fulfilling the processor's mapping requirements [9].

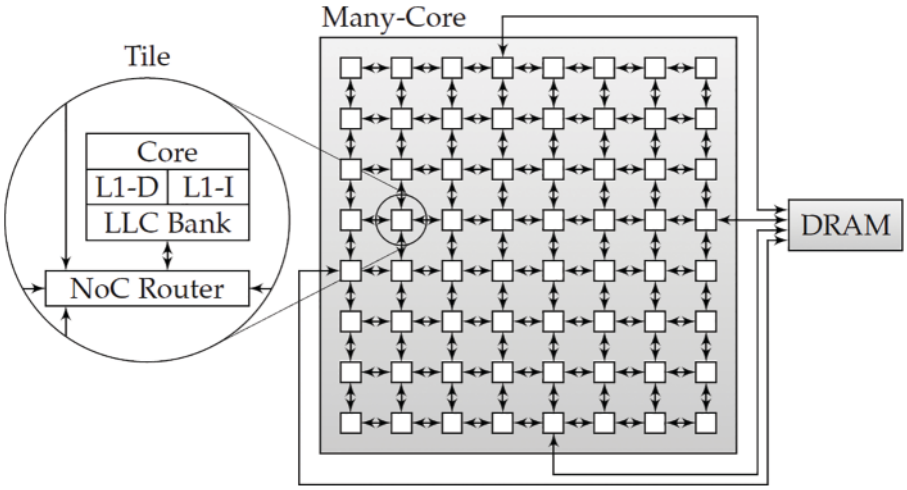


**Figure 1:** Tasks scheduling based on LLC using CNN

### *1.1 Dissipation in MPSoCs*

Energy dissipation and Energy-Efficient Scheduling play a vital role in the Migration of tasks while using a Homogeneous platform to meet deadlines of tasks during execution [10]. An energy-efficient dissipation method is proposed to enhance the task migration abilities of MPSoC. Introduce an efficient energy dissipation strategy. Shrinkage of the chip and certainly increasing the electronic element transistor on a chip the frequency and power densities are gradually increasing causing many problems like power consumption and thermal issues as well as higher dissipation of energy [11]. Efficient multi-core systems are introduced but due to the increase in energy thermal issues arise that are the major problems [12].

Introduces a mechanism that energy consumption ($E_{cc}$) of executing a task on an MPSoC mainly by switching resources [13] as shown below in using Eq. (1):

$$E_{cc} = P_{switching}.t = C_{eff}.V^2.f\frac{C_b}{f} = C_{eff}.V^2.C_b(V_{dd} - V_{ths})/.{V_{dd}}^2 \tag{1}$$

In CMOS chips, the consumption of dynamic power due to the transistor's switching can be calculated using Eq. (2):

$$P_{dy} = C_l * N_c * V_d * f \tag{2}$$

$$f = M * (V_d - V_t)/{V_d}^2 * f \tag{3}$$

Slave and Master thread Migration ratio in AI-based MPSoC that shows core performance using AMD hops in Fig. 2. The load capacitance is denoted as $C_l$, $V_d$, $N_c$ is voltage and no of state transitions. The clock frequency is represented as $f$ while $V_t$ represents the threshold voltage, $L_g$ shows the logic gates used in the CMOS chip. The reduction constant is denoted as $M$ as shown in Eq. (3). The accumulative power of the CMOS is shown in Eq. (4):

$$P_t = (C * {V_d}^2 f + L_g * (V_d * M_3) \tag{4}$$

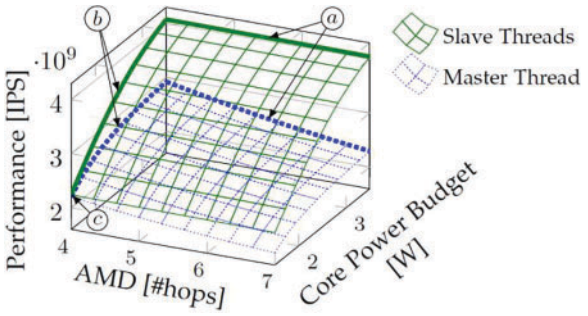


**Figure 2:** Slave and master thread migration ratio in AI-based MPSoC

### *1.2 Task Migration in MPSoC*

Task migration based on GPU memory for power efficient MPSoC is a mechanism that is used to move an executing task from one host CPU in a distributed architecture to another CPU. DVFS based selection of a task and movement to the host CPU for a new time can task and the creation of the task on that host increase the performance, reliability and processing speed. The lack of task migration on affect the overall system performance [14]. Most of the current techniques improve energy and power management-related

problems. The mapping of ready task allocation to CPU is another approach to scheduling, there are three main classes of task scheduling multiprocessor partitioned scheduling, restricted-migration scheduling, and full-migration scheduling.

### *1.3 Partitioned Migration Scheduling*

Introduced an energy aware partitioned migration scheduling technique that is used for the optimization of energy-aware distributed multiprocessor. The system mapped each ready task to a single processor $\pi$ at the system-design time [15]. Statically assignment of the task to the CPU is preferable for the uniprocessor schedule as illustrated in Fig. 3.

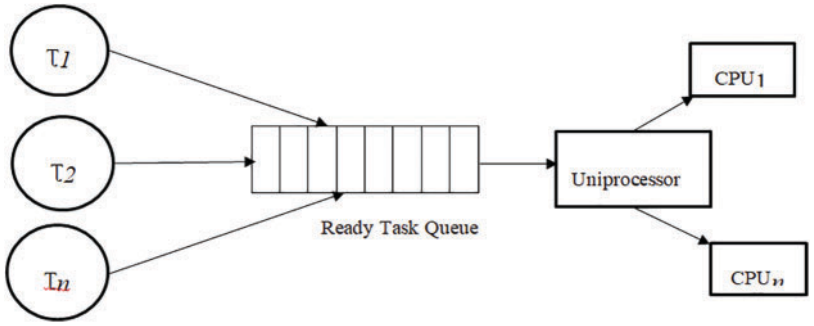


**Figure 3:** Tasks scheduling using partitioned migration

### *1.4 Full Migration Scheduling*

Introduced full migration task scheduling technique based on dynamic voltage and frequency scaling DVFS that is widely used as the least restrictive method widely used in scheduling. An energy and performance efficient DVFS scheme can handle task that halt on one CPU and reschedule the execution on a different processor to avoid delay and meet the deadline. Job-level parallelism is restricted because tasks cannot execute parallel on two or more different CPU cores [16]. Each arrived task is placed into a priority queue. The scheduler can have the option to decide what task requires execution on each CPU at the current time interval as shown in Fig. 4.

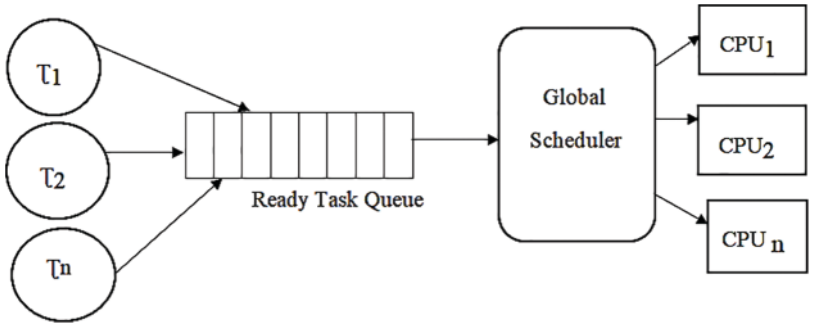


**Figure 4:** Tasks scheduling of tasks using partitioned migration

### *1.5 Restricted Migration Scheduling*

Introduce a restricted migration scheduling based on a Resource-aware load balancing model for a batch of tasks (BoT) considering the deadline and best-fit migration policy for homogenious and heterogeneous distributed computing systems that migrate tasks between CPUs, each arrived task can execute on only one CPU core at a time [17]. Proposed a Dynamic scheduling technique for real-time tasks in heterogeneous multicore systems in which the arrivals of the task are not prioritized. In restricted migration, a scheduler can adapt to two levels. Generated tasks are placed on the global priority queue [18]. Reference [19] introduced an energy optimization for real-time multiprocessor that runs only periodic tasks with uncertain execution time by considering the migration policy to reduce the effects of the increase in temperature on the chip that can smoothly perform and achieve a normal working condition. A uniprocessor scheduling algorithm is used to schedule each processor's assigned task as shown in Fig. 5.

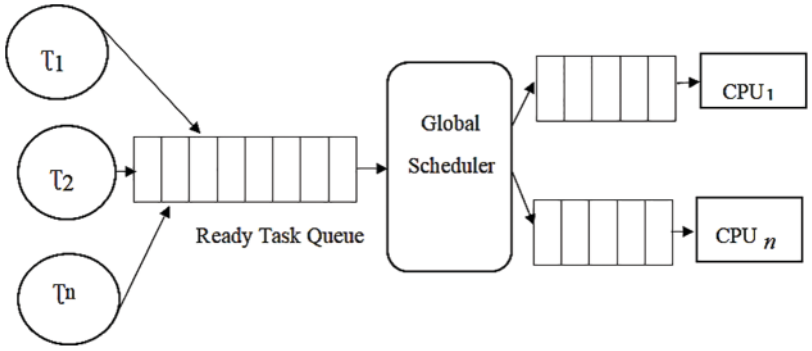


**Figure 5:** Tasks scheduling using partitioned migration

## 2 Literature Review

System-level application-aware dynamic power management (DPM) in adaptive pipelined MPSoCs for software-based energy and power reduction schemes are implemented at the consumer level and the consumption of energy is reduced using the software. Instruction reordering and energy-efficient code are widely used software-based energy optimization techniques. Energy optimization using software-based methods includes instruction level, operating system (OS) level and compiler level implementation [20]. Introduce an Online dynamic power management (DPM) for efficient execution of interactive workloads AND instruction-level energy optimization technique. The instruction level analysis is used to assign a defined energy budget to any ready task. Proposed a compiler-level mechanism that plays an important role in the behavior of an application during execution and determines the no of instructions being executed, order as well as type and has a significant impact on the reduction of energy consumption [21]. Dynamic power management (DPM) for multidomain system-on-chip is introduced with an optimal control approach to reduce the consumption of energy and power. Introduces a memory latency optimization technique that is used for the latency optimizations at the compiler level That modifies the memory layout and program access pattern operating system (OS) level [22]. The implementation of Optimal Dynamic power management for multidomain is used for the optimization of software-level approach and also used in hardware as pointer synthesis [23].

Proposed a real-time model that is based on finite elements and used for the optimization and reduction of memory by considering dynamic memory and dynamic power approach as high-speed on-chip memory is widely used to reduce the consumption of energy during the run time. Introduce an

instruction rescheduling technique that is highly used compiler-level energy and power reduction technique that is used to enhance the performance as well as also remove the pipeline stalls that cause a delay in processing [24]. Proposed an analysis and optimization of MPSoCs reliability test that utilize the operating system according to the requirement and demand of tasks related to OS processes. The OS level optimization technique is implemented on dynamic power management (DPM), CPU clock, scheduling algorithms, Input-output (I/O) and speed management software-based DPM technique using an operating system level is implemented to reduce energy for performance enhancement as shown in Eq. (5).

$$E_T = E_s + E_p + E_C \tag{5}$$

The energy consumed during the transmission is represented as $E_{Transmission}(n_i, d)$ bits can be calculated using Eq. (6).

$$E_{Transmission}(n_i, d) \geq \begin{cases} n_i * E_{ss} + n_i * e_{ffs} * \left(d^2\right) & d \leq d_o \\ n_i * E_{ss} + n_i * e_{ffp} * \left(d^4\right) & d \geq d_o \end{cases} \tag{6}$$

Whereas $E_{ss}$ represents the energy dissipation per bit in the transmitter circuit $e_{ffp}$ and $e_{ffs}$ are the amplification energy [25]. Due to advancements in embedded technology, periodic activities show more computational demand in a multi-core system these activities are categorized as periodic task sets is continuously executed at definite rates with non-elastic deadlines. when the CPU is not working and is in an idle state then the maximum duration can be computed using λ so that $t_i$ with the earliest arrival $\alpha_k$. When another task with the nearest deadline is executed, the $t_i$ can be delayed using Eq. (7).

$$\sum_{i=\{1.,.,.n\}/k}^{n} \frac{c_i}{\tau_i} + \frac{c_k + \alpha_k}{\tau_k} = 1 \tag{7}$$

While another task $t_2$ with higher priority arrive with the earliest deadline before the end of the execution task $t_1$ then the length of the idle interval is denoted as $\lambda_j$ and max time duration for the idle period is represented as $\alpha_j$ can be measured using Eq. (8).

$$\sum_{i=\{1.,.,.n\}/(k,j)}^{n} \frac{c_i}{\tau_i} + \frac{c_k + \alpha_k}{\tau_k} + \frac{c_j + \alpha_j}{\tau_j} = 1 \tag{8}$$

The utilization bound $\propto$ is a function of $\frac{1}{u_{max}(A)}$ scheduling of n tasks on m identical CPUs. If $n \leq \propto m$ then the CPU will be allocated to the task set while in case $\geq \propto m$ then $\propto$ the utilization bound for P-EDF on the CPU can be calculated using Eq. (9) is considered as:

$$u_{sum} \leq \propto m + 1/\propto + 1 \tag{9}$$

Introduce a P-EDF-based utilization bound for identical MPSoC first fit decreasing (FFD), best fit (BF) and random fit (RF) are widely used for assigning tasks to CPU and it's observed that all these variations have the same utilization bound. Introduce a mechanism to find the utilization bound that determines the least number of processors that are required to schedule $n$ tasks with $u_{sum}$ as shown below in Eq. (10).

$$m \geq \begin{cases} 1 \\ \min\left\{\left\lceil \frac{n}{\propto} \right\rceil \right. \end{cases} \quad if\ u_{sum} \leq 1 \tag{10}$$

Marvel X-Scale AMD PXA-270 is widely used for symmetric multiprocessing when a real-time task load arrives and all the processors in a multi-core embedded system are of the same type. Simulation tools for real-time multiprocessors (STORM) implement various scheduling techniques on MPSoC-based architecture using both homogeneous and heterogeneous CPUs. STORM generates the energy profiles and evaluates performance and task scheduling on the MPSoC [26].

## 3 An Improved Machine Learning-Based EA-EDF Scheduling Model

The scheduling algorithm applies migration of task $t_i \in \tau$ when a multiprocessor system allows a $t_i \in \tau$ migrates if it starts execution on one core of the processor and later switches due to high utilization of the task and migrates the task to the other core.

Using the proposed full task migration strategy in which all the task $t_i \in \tau$ are allowed to migrate at any point as per the condition of $u_i$ implemented in the proposed EA-EDF during their execution. However, the $t_i \in \tau$ cannot execute parallel on multiple cores, it can execute on any single core selected from the core configuration at a time. This strategy is the most flexible and increases performance by allowing migration. Fig. 6 illustrates the proposed machine Learning-based task migration Model by integrating CNN into EA-EDF full migration of tasks using EA-EDF. The task set contains a ready task set that is allocated to the scheduler the proposed scheduler monitors the utilization factor and migrates the $t_i$ according to the configurations.

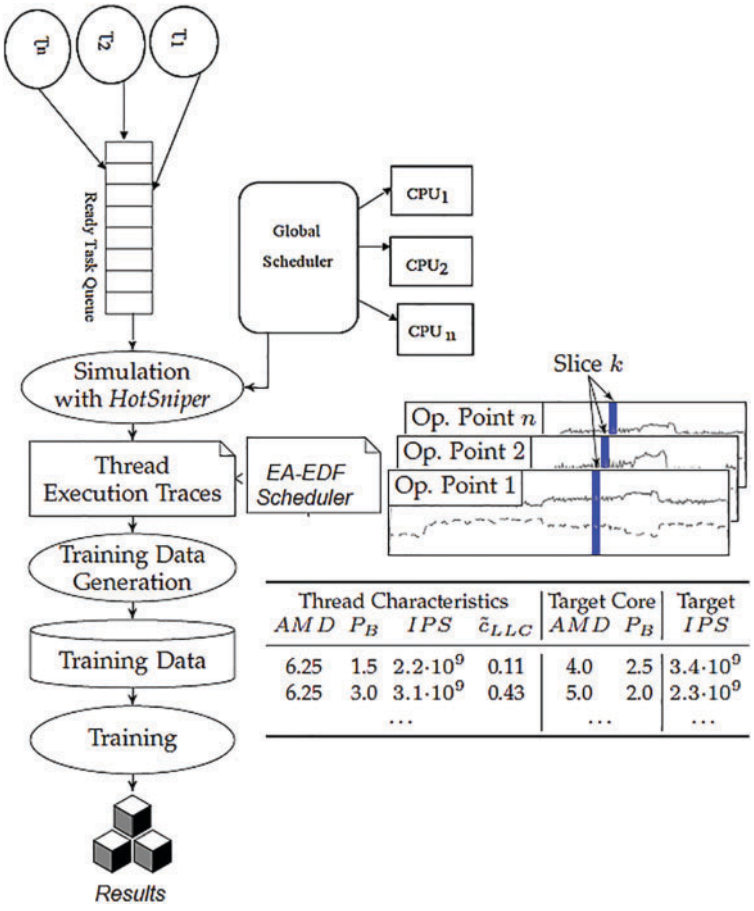


**Figure 6:** Machine learning-based task migration model using integrating CNN into EA-EDF

Considering the concurrent arrival of tasks having the same deadlines and priority and the task required the CPU cycle for mapping and allocation of resources access to CPU. Improper task scheduling consumes more energy and causes a delay in the CMOS chip. High task utilization without proper allocation of CPU degrades the reliability and performance of MPSoC, missing task $(\tau)$ deadlines particularly periodic ready task set $\tau = (\tau_1, \tau_2, \tau_3, \ldots, \tau_n)$ and scheduling cause a 10%–15% delay in CMOS circuits due to improper task migration between symmetric and asymmetric cores over $m$ cores multiprocessor. When two processes require a CPU cycle at the same interval increases the CPU load and energy dissipation. Completion of a task by meeting the task deadlines and by allowing the suitable task migration policy and DPM-based core switching technique can reduce the consumption of energy and enhance the overall performance. The design issue of a predictable and energy-efficient migration of processes in shared-memory MPSoCs has not received enough attention. The objective of our research work presented in this article is to propose a more mature and improved task migration solution that supports the processor to reduce the dissipation of energy and increase the overall performance.

***Dataset Description and Preprocessing***

The PTB Suggestive ECG Data set, which comes from Physionet, is the dataset that was used in this research for processing over the CPU as a ready task the dataset is publically open and available for use as referred to in [27]. The 14,552 ECG accounts in this dataset are isolated into two packs are used during the testing and training process because using more than 14,552 samples can cause a very high upsurge and requires to make atlest three packs for the ML system in our upcoming research with more advanced ML system we will be able to use 21,837 ECG samples together. Considering two uniform multiprocessors denoted by $P_1$ and $P_2$ and a (R-T) system instance, $I = \{t_1, t_2, \ldots, t_n\}$. Let $s_{p(1)}$ and $s_{p(2)}$ represent the schedule of instance $t_1, t_2, \ldots, t_n$ on $P_1$ and $P_2$ respectively when the workload increases the tasks migrate to the core in the selected core configuration that is the least used then for instance $I$ and time instant $t$. Consider a task set $t_i \in \tau$ $(t_1, t_2, \ldots . t_n)$ and sort ready task queue $t_i \in \tau$. The migration of tasks occurs instantly when $u_i \geq$ max allowable workload. We have used a work function for MPSoC to elaborate the total average work done by all the tasks $t$ of instance $I$ over-scheduled time intervals $t > 0$. The system work function is represented as $W(s_{p(1)}, P, I, \tau) \geq W(s_{p(2)}, P, I, \tau)$ migrates tasks on the core when $u_i \geq$ threshold workload without any delay by meeting all the deadlines.

Let's assume the parameters $(r_k, c_k, d_{k,} p_{k,} p_{rk})$ of tasks $t_i \in \tau$ be a task in $RT$ instance $\{t_k, \ldots, t_n\}$ at counter time $t_o$ Set the power parameter $P_{sleep}(\beta), P_{idle}(\Upsilon)$ and $P_{running}(\alpha)$ of AMD PXA-270 multiprocessor. Consider a task $t_k \in \tau$ with the earliest arrival is ready for allocation to core configuration $(\lambda_1, \lambda_2, \ldots \lambda_n)$ if $(u_i) <$ max allowed $\tau$ load than the work function is represented as shown below in Eq. (11).

$$W(s_{p(1)}, P_1, I, t_k, t_o)\, W(s_{p(2)}, P_2, I, t_k, t_o) \tag{11}$$

Eq. (11) states that initial. $u_i <$ *max allowable workload* $(T)$. Considering the utilization for the whole instant $I$ as $(t_k \in \tau) \in I$ to calculate the overall response as shown below in Eq. (12).

$$W\left(s_{p(1)}, P_1, I, t_o\right) < W(s_{p(2)}, P_2, P, I, t_o) \tag{12}$$

Eq. (13) illustrates if $t_o = r_{k,}$ then the work done on $P_1$ for instance, I will increase as the arrival of the new task $t_k$ increases the overall utilization of the selected configuration. If $(u_i) \geq$ max allowed $\tau$ load, $rk$, is the peak task load period.

$$W(s_{p(1)}, P_1, I, r_{k,}) \geq W(s_{p(2)}, P_2, I, r_{k,}) \tag{13}$$

Therefore, the completion of $I = \{t_k, \ldots, t_n\}$ of $RT$ instance schedule in $s_{p(1)}$ during the time interval $[t_o, r_{k,})$ is greater than the schedule $s_{p(2)}$ during the same time interval. In processor schedule $s_{p(1)}$, the task $t_k$ is active during the entire interval $[t_o, r_{k,})$ so at least two processors will be running during that time interval and the remaining will be in an idle state as shown below in Eq. (14).

$$t_o - r_k = a_1 + a_2 + \ldots + a_m(p) \tag{14}$$

If $t_k$ schedules on $P_2$ at $[t_o, r_{k,})$ and $P_1$ is already at its threshold utilization than using the proposed task migration framework strategy the arrival of any new task $\boldsymbol{t_k} \in \boldsymbol{\tau}$ schedules and starts executing on the next available processor or core in the selected core configuration $(\boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, \ldots \boldsymbol{\lambda_n})$ that is in the least use according to the $u_i$ therefore it can be expressed as shown below in Eq. (15).

$$W(S_o, P_2, I, r_k, t_o) \leq s_{p(1)}(P_2) * (t_o - r_k) \tag{15}$$

Task $t_k$ can schedule and start executing when the $u_i$ reaches its threshold If $(u_i) \geq$ max allowed $\tau$ load enables the $\tau$ migration using the proposed EA-EDF. A minimum of one processor in the core configuration $(\lambda_1, \lambda_2, \ldots \lambda_n)$ must be in an idle state to schedule the task. In case all the processor $m(p)$ of $P$ are busy then the task $t_k$ has to wait till the availability of one processor. Therefore $t_k$ is guaranteed to be scheduled without missing its deadline for the set of unit time intervals $(\sum_{k=1}^{m(P)-1} a_k)$.

Moreover in the $c_k$ WCET, $t_k$ Executes on the idle or slowest processor. When a task $t_k \in \tau$ are executing some jobs on $s_k(P_m)$ Therefore Select the core configuration $(\boldsymbol{\lambda_n})$ as per the $(\boldsymbol{u_i})$ requirement of arrived $\boldsymbol{t_i} \in \boldsymbol{\tau}$, either least used $(\boldsymbol{\lambda_n})$ as shown below in Eq. (16).

$$W(S_o, P_2, I, t_k, t_o) \geq \sum_{k=1}^{m(P)-1} a_k s_k(P) \tag{16}$$

CPU core at the speed of $k$th the processor requires the switching of the processor's $P_{sleep}$ $(\beta)$ state to $P_{idle}$ $(\Upsilon)$ in core configuration $(\lambda_n)$ Switching $(\lambda_n)$ from $P_{sleep}$ $(\beta)$ to $P_{idle}$ $(\Upsilon)$ to achieve optimal utilization of the core as the instant $(\sum_{k=1}^{m(P)} a_k s_k (P_m) = \delta$ is a higher utilization task than $\delta(P_m)$ must be less than the load for proper scheduling of tasks. Based on the above equation it's confirmed that using unrestricted full migration permits all the tasks to migrate among processors in the configuration if required.

The proposed framework migrates tasks to the core when $u_i \geq$ threshold workload without any delay by meeting all the deadlines and satisfying the average work done by all the tasks $t$ of instance I over scheduled time intervals t > 0: The below mentioned are the various $m$ cores distribution models selected during the $t_i \in \tau$ workload on $u_i$ 12%, 31%, and 50% using the proposed task scheduling method. CPU in the sleep state is represented as $P_{sleep}$ $(\beta)$ While the CPU in idle and executions state is denoted as, $P_{idle}$ $(\Upsilon)$ and $P_{running}$ $(\alpha)$ at various standard frequencies, 624, 208 and 104 MHz of MARVEL AMD PXA-270 embedded MPSoC whereas the no of the core is represented as $m_n$.

Eq. (17) represents the integration of a full task migration model for various utilization factors $(u_i)$ = 12% into EA-EDF scheduling algorithm for mapping and migration of arrived task under minimal utilization when the CPU $\alpha = 2, \beta = 6$ considering the mapping core distribution as $\lambda_2$: $(m_1, m_2)$, $(m_4, m_7)$, $(m_3, m_5)$, $(m_6, m_8)$.

$$u_\tau = \sum_{K=2}^{n} \frac{c_a}{p_a} = 12\% \; U_\tau \leq m(p) + 2 \tag{17}$$

Eq. (18) represents the integration of a full task migration model for various utilization factors $(u_i)$ = 31% into EA-EDF scheduling algorithm for mapping and migration of arrived task under minimal utilization when the CPU α = 4, β = 4 considering the mapping core distribution as $\lambda_4$: ($m_1$, $m_2$, $m_3$, $m_4$), ($m_5$,$m_6$, $m_7$, $m_8$).

$$u_\tau = \sum_{K=4}^{n} \frac{c_a}{p_a} = 31\% \quad U_\tau \le m(p) + 4 \tag{18}$$

$$u_\tau = \sum_{K=6}^{n} \frac{c_a}{p_a} = 50\% \quad U_\tau \le m(p) + 6 \tag{19}$$

Eq. (19) represents the integration of a full task migration model for various utilization factors $(u_i)$ = 50% into EA-EDF scheduling algorithm for mapping and migration of arrived task under minimal utilization when the CPU α = 6, β = 2 considering the mapping core distribution as $\lambda_6$: ($m_1$, $m_2$, $m_3$, $m_4$, $m_5$, $m_7$), ($m_1$, $m_2$, $m_3$, $m_6$, $m_7$, $m_8$). Fig. 7 illustrates the various core configuration states of the proposed $m$ cores distribution model using a full task migration policy for $t_i \in \tau$ on MARVEL AMD PXA-270 embedded MPSoC based on utilization factor $(u_i)$.

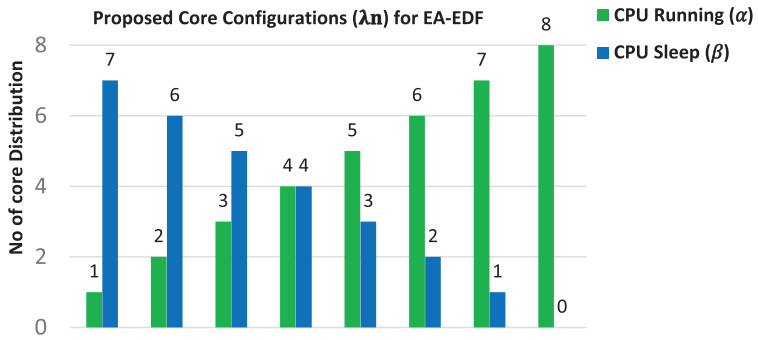


**Figure 7:** Various proposed stages for the core distribution mode $(\lambda_n)$ based on $(u_i)$

## 4 Simulation and Results

The experimental evaluation of uniform multiprocessor is considered using STORM for the multi-threaded application that executes an XML file containing the task (τ) parameters for $t_i \in \tau$, $n$ = 4, $n$ = 10 and $n$ = 17. Fig. 8 represents the core distribution model for the X264 CPU core in which the period ranges between [4–25 ms]. The task set in Tables 1–3 on $u_i$ = 12, $u_i$ = 31, $u_i$ = 50% were evaluated on $s_d$ = 1000 ms simulation duration for embedded architecture supporting symmetric as well asymmetric CPU arrangement with $m$ CPUs. Task $t_i \in \tau$, with $u_i$ = 12%, $u_i$ = 31% and $u_i$ = 50% are allocated and mapped on the CPU cores ($\lambda_2$,$\lambda_4$, $\lambda_6 \in \lambda_n$).

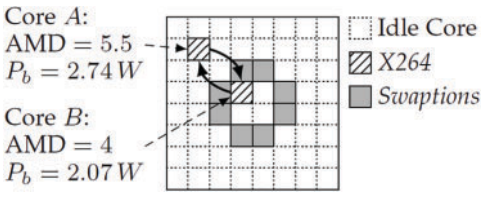


**Figure 8:** Core distribution model for X264 CPU Core

**Table 1:** Parameters for system timing requirements (ms), at $u_i$ = 12% for task set $\tau_n$ = 4, $m$ = 2

| *Tasks* $\tau$ | $p_i$(ms) | $r_i$ (ms) | $d_i$ | $c_i$ (ms) | $p_r$ | *Tasks* $\tau$ | $p_i$(ms) | $r_i$ (ms) | $d_i$ | $c_i$ (ms) | $p_r$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $t_1$ | 4 | 4 | 4 | 2 | 7 | $t_3$ | 16 | 11 | 16 | 2 | 5 |
| $t_2$ | 15 | 10 | 15 | 2 | 8 | $t_4$ | 15 | 8 | 15 | 3 | 6 |

**Table 2:** Parameters for system timing requirements (ms), at $u_i$ = 31% for task set $\tau_n$ = 10, $m$ = 4

| *Tasks* $\tau$ | $p_i$(ms) | $r_i$ (ms) | $d_i$ | $c_i$ (ms) | $p_r$ | *Tasks* $\tau$ | $p_i$(ms) | $r_i$ (ms) | $d_i$ | $c_i$ (ms) | $p_r$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $t_1$ | 10 | 0 | 10 | 1 | 1 | $t_6$ | 6 | 4 | 6 | 2 | 5 |
| $t_2$ | 9 | 2 | 4 | 2 | 6 | $t_7$ | 9 | 2 | 9 | 1 | 1 |
| $t_3$ | 4 | 4 | 20 | 1 | 7 | $t_8$ | 4 | 4 | 4 | 2 | 6 |
| $t_4$ | 15 | 0 | 40 | 1 | 8 | $t_9$ | 15 | 0 | 15 | 1 | 7 |
| $t_5$ | 17 | 2 | 20 | 2 | 5 | $t_{10}$ | 19 | 2 | 17 | 1 | 10 |

**Table 3:** Comparison of energy dissipation at various ($u_i$) and core state ($\alpha, \beta$) at 624 MHz

| Utilization % & core state | Task $\tau$ | Proposed EA-EDF | GEDF | PDTM | Uniform RT-DPM | TBP |
|---|---|---|---|---|---|---|
| $\forall\tau \rightarrow u_\tau = \sum_{m=1}^{n} = 7, \alpha = 1\,\beta = 7$ | 2 | 0.93 | 1.41 | 1.18 | 0.86 | 1.25 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=2}^{n} = 11, \alpha = 2\,\beta = 6$ | 8 | 0.97 | 1.81 | 2.02 | 1.27 | 1.87 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=3}^{n} = 13, \alpha = 3\,\beta = 5$ | 11 | 1.12 | 2.73 | 1.99 | 1.98 | 3.21 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=4}^{n} = 22, \alpha = 4\,\beta = 4$ | 14 | 1.71 | 3.2 | 3.91 | 3.52 | 3.79 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=5}^{n} = 39, \alpha = 5\,\beta = 3$ | 18 | 2.86 | 6.12 | 4.37 | 4.12 | 4.88 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=6}^{n} = 53, \alpha = 6\,\beta = 2$ | 22 | 3.86 | 7.23 | 6.32 | 6.44 | 5.21 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=7}^{n} = 60, \alpha = 7\,\beta = 1$ | 28 | 4.19 | 8.26 | 6.38 | 5.76 | 6.96 |
| $\forall\tau \rightarrow u_\tau = \sum_{K=8}^{n} = 64, \alpha = 0\,\beta = 8$ | 33 | 8.69 | 8.78 | 8.86 | 8.90 | 9.22 |

All the tasks are periodic and denoted as $\tau = (t_1, t_2, \ldots, t_n)$ using the hardware and software architectural parameters of Marvel X-Scale AMD PXA-270 MPSoC based on the proposed full migration and scheduling policy with dynamic power management (DPM) integration based on the utilization factor ($u_i$) of the CPU core with the same energy and power for all cores are the same. Table 1 represents the parameters of system timing requirements.

Table 2 represents the parameters of system timing requirements at 31% utilization factor considering 4 core CPU.

Fig. 9 represents the analysis that compares the energy dissipation of the proposed approach on numerous frequencies using AMD PXA-270. Table 3 represents the comparison of energy dissipation on various clock frequencies underutilization. $u_i$ = 6%, 10%, 20%, 36%, 55%, 60%, 62.5% and $u_i \geq 63\%$ when all the cores are running. Simulation shows that EA-EDF gives more energy-efficient results at lower ($u_i$) utilization.

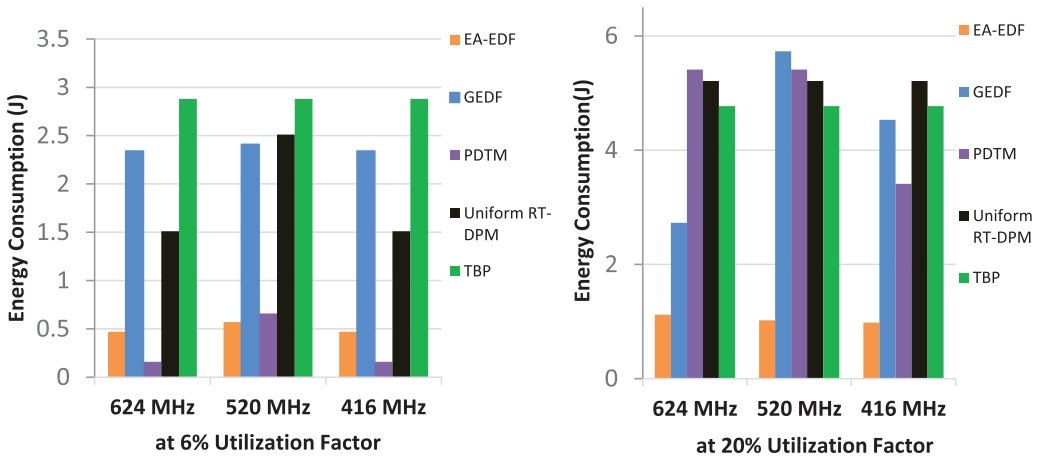


**Figure 9:** Comparison of energy consumption at $u_i$ = 6%, 20%, 31% and 53% at 624, 520 and 416 MHz

Figs. 10 and 11 represent the analysis that compares the performance of our proposed approach EA-EDF using AMD PXA-270 MPSoC in terms of dynamic energy dissipation for a different ready task set at various $u_i$ utilization factors with other conventional techniques like Multi-Layer Feed-Forward Neural Network (MLFNN) based on convolutional neural network (CNN), Random Forest (RF) and Deep learning (DL) algorithm at 624 and 416 MHz. The $X$-axis is representing the proposed EA-EDF $u_i$ utilization factor in comparison to other heuristic techniques increasing with the arrival of a new task $t_i \in \tau$; $Y$-axis represents the improvement % due to efficient task migration and DPM-enabled policy improving the performance by reducing energy dissipation.

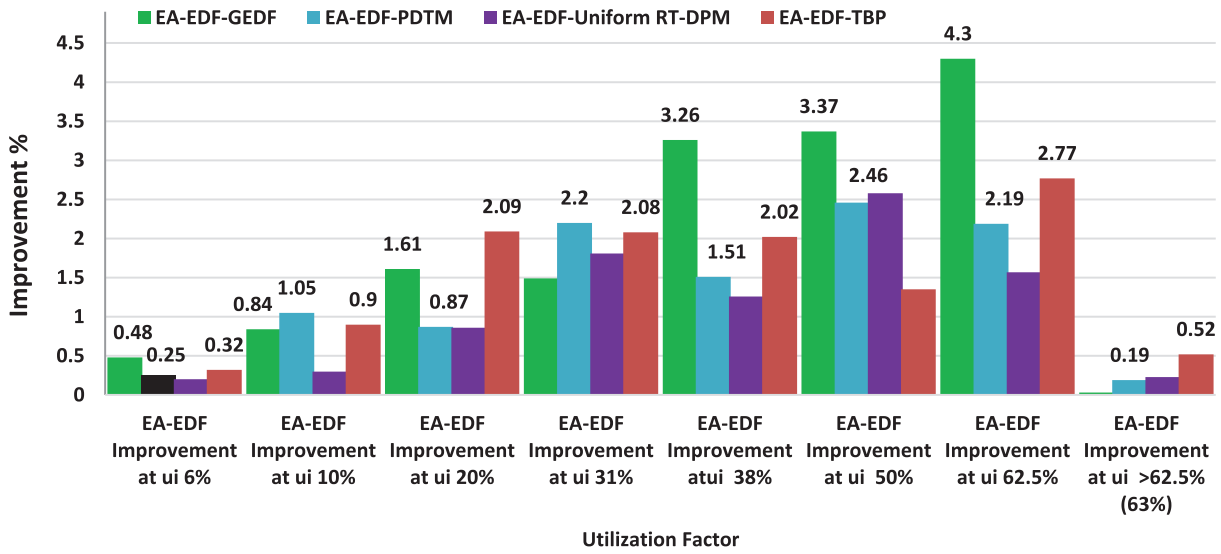


**Figure 10:** Proposed EA-EDF energy consumption comparison with G-EDF, PDTM U-RT-DPM, TBP at 624 MHz

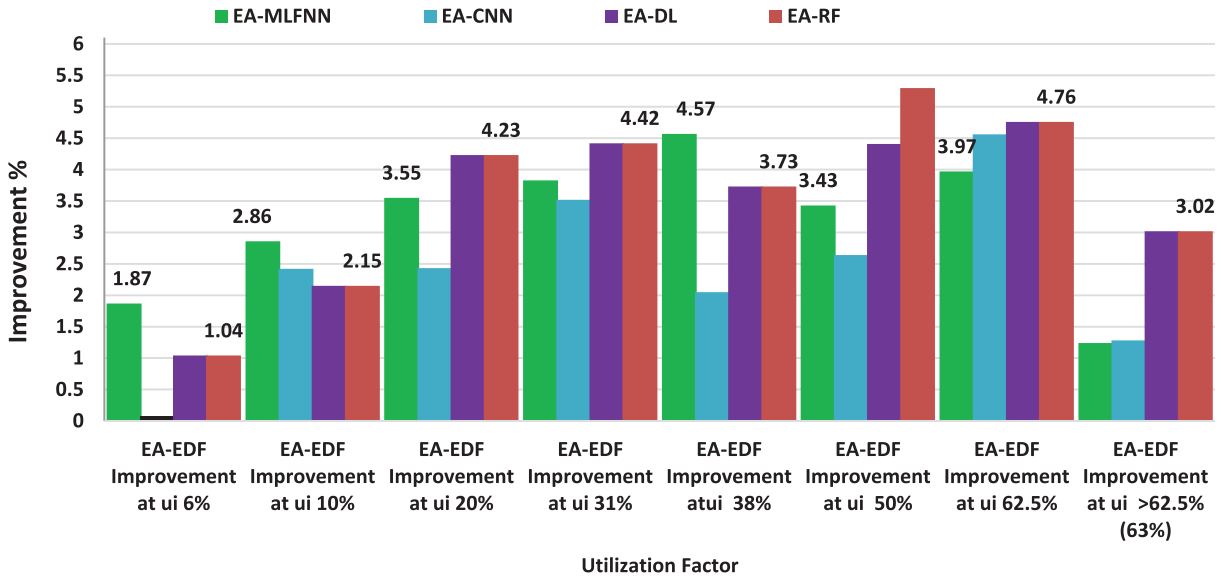


**Figure 11:** Proposed EA-EDF energy consumption comparison with EA-MLFNN, EA-CNN, EA-DL-EA-RF at 416 MHz

DPM is more convenient when tasks $t_i \in \tau$ increases, the task starts migration and the shorter slack time intervals are optimized and move to an idle state due to the DPM approach once the heuristic starts executing the average energy is computed by considering that the release time of the task is its first arrival and each processor $m$ with running states need to be allocated. The Simulation results show the maximum improvement. The average and the minimum improvement of our proposed approach compared to the GEDF approach are 4.3%, 2.25% and 0.48%, respectively. Proposed ML-based EA-EDF gives max improvement at ($u_i$ = 38%) on m = 5 running processor, while on average improvement occurs at ($u_i$ = 50%) on m = 6 running processor and the least improvement occurs at ($u_i$ = 6%) on m = 1 running the processor in an 8-processor platform. So we can generally state that when the $u_i$ utilization factor is higher due to a higher number of task executions making the dynamic power management technique more effective that's why we can see a significant improvement at ($u_i$ = 62%) The overall improvement is 4.3%, 1.27%, 0.26% and 1.02% on ($u_i$ = 50%) 3.37%, 1.46%, 0.58% and 0.35% in comparison with G-EDF, PDTM U-RT-DPM, and TBP utilization.

## 5 Conclusion

This article effectively demonstrates the viability of utilizing Convolutional Neural Network (CNN) based Multiprocessing architectures that are used for high-dimensional datasets, especially with available electrocardiography datasets to evaluate the task scheduling and migration that play a vital role in energy reduction, particularly in multiprocessor design. The rapid increase in the demand for multithreaded applications enhances the thermal heat dissipation in MPSoC. A full task migration policy based on Dynamic power management techniques (DPM) for efficient task allocation and scheduling is the finest integration used for decreasing the consumption of energy. MARVEL AMD PXA-270, PXA-271 and PXA-250, are using system-level DPM for switching CPU modes as per the need of the energy-power model. Various parameters like WCET, BCET, deadline, period and priority are the main characteristics of multithreaded real-time (R-T) applications. In this article, we have designed a system-level task migration policy used to tackle improper task allocation to the CPU core and inefficient task scheduling for real-time applications with deadline and priority constraints that ultimately reduce energy dissipation and improve overall performance.

**Acknowledgement:** The authors sincerely thank **Prof. Son Lam Phung** from the University of Wollongong, Australia for important suggestions and feedback on this paper.

**Funding Statement:** The authors received no specific funding for this study.

**Author Contributions:** Hamayun Khan is the corresponding author of this article, he worked on data analysis, the introduction and methodology of research paper by explaining the scope, context, research from write-up to analysis contributed critically to the research paper. Muhammad Atif Imtiaz worked on the development of the algorithm and results analysis in depth. Hira Siddique worked on the development of mathematical analysis of the research work and aligned the algorithm and flow chart analysis from implementation to results initiation. Arshad Ali worked on the proofreading and implemenation of the dataset, testing, and validation of the proposed model. Muhammad Zeeshan Baig worked on literature and data collection processes that helped in the implementation of the proposed model and migration policy using three-stage migration techniques in the research. Muhammad Tausif Afzal Rana validated the methodology and mathematical analysis by evaluating the model on STORM tool and generated high-resolution results for migration policies. Saif ur Rehman developed all the tables and calculated the utilization factor at various frequencies, especially implementation at 512 higher frequencies. Yazed Alsaawy's role is to enhance the overall structure of the research paper by removing the typo mistakes supported in proofreading and English grammar enhancing the overall quality of work. All authors reviewed the results and approved the final version of the manuscript.

**Availability of Data and Materials:** All data are available from the corresponding author on reasonable request.

**Ethics Approval:** Not applicable.

**Conflicts of Interest:** The authors declare no conflicts of interest to report regarding the present study.

## References

1. Irwin MJ, Benini L, Vijaykrishnan N, Kandemir M. Techniques for designing energy-aware MPSoCs. Multiprocess Syst-on-Chips. 2005;13(5):21–47. doi:10.1016/B978-012385251-9/50016-5.
2. Burns A, Andy W. Dispatching domains for multiprocessor platforms and their representation in ada. In: International Conference on Reliable Software Technologies; 2010; Berlin/Heidelberg, Germany: Springer. p. 41–53.
3. Gonzalez-Martinez G, Sandoval-Arechiga R, Solis-Sanchez LO, Garcia-Luciano L, Ibarra-Delgado S, Solis-Escobedo JR, et al. A survey of MPSoC management toward self-awareness. Micromachines. 2024;15(5):577. doi:10.3390/mi15050577.
4. Liu J, Mao M, Gao J, Bai J, Sun D. Hardware-accelerated YOLOv5 based on MPSoC. J Phys: Conf Ser. 2024;2732(1):012013. doi:10.1088/1742-6596/2732/1/012013.
5. Verma P, Maurya AK, Yadav RS. A survey on energy-efficient workflow scheduling algorithms in cloud computing. Softw Pract Exp. 2024;54(5):637–82. doi:10.1002/spe.3292.
6. Yu T, Zhong R, Janjic V, Petoumenos P, Zhai J, Leather H, et al. Collaborative heterogeneity-aware OS scheduler for asymmetric multicore processors. IEEE Trans Parallel Distrib Syst. 2021;32(5):1224–37. doi:10.1109/TPDS.2020.3045279.
7. Rao W, Li H. Energy-aware scheduling algorithm for microservices in Kubernetes clouds. J Grid Comput. 2024;23(1):2. doi:10.1007/s10723-024-09788-w.
8. Gaffour K, Benhaoua MK, Benyamina AEH, Singh AK. A new efficient multi-task applications mapping for three-dimensional network-on-chip based MPSoC. Concurr Comput Pract Exp. 2021;33(10):1–20. doi:10.1002/cpe.6194.
9. Hu Y, Liu Y, Liu Z. A survey on convolutional neural network accelerators: GPU, FPGA and ASIC. In: 2022 14th International Conference on Computer Research and Development (ICCRD); 2022 Jan 7–9; Shenzhen, China: IEEE; 2022. p. 100–7. doi:10.1109/ICCRD54409.2022.9730377
10. Gonzalez R, Horowitz M. Energy dissipation in general purpose microprocessors. IEEE J Solid State Circuits. 1996;31(9):1277–84. doi:10.1109/4.535411.

11. Choi S, Prasanna VK, Jang JW. Minimizing energy dissipation of matrix multiplication kernel on Virtex-II. In: Reconfigurable technology: FPGAs and reconfigurable processors for computing and communications IV. Boston, MA, USA: SPIE; 2002. doi:10.1117/12.455487.
12. Ali H, Tariq U, Hardy J, Zhai X. A survey on system level energy optimisation for MPSoCs in IoT and consumer electronics. Comput Sci Rev. 2021;41(1):100–16. doi:10.1016/j.cosrev.2021.100416.
13. Feliu J, Sahuquillo J, Petit S, Eeckhout L. Thread isolation to improve symbiotic scheduling on SMT multicore processors. IEEE Trans Parallel Distrib Syst. 2020;31(2):359–73. doi:10.1109/TPDS.2019.2934955.
14. Somdip D, Isuwa S, Saha S, Singh AK, Maier KM. CPU-GPU-memory DVFS for power-efficient MPSoC in mobile cyber physical systems. Future Internet. 2022;14(3):91–103. doi:10.3390/fi14030091.
15. Jiang E, Wang L, Wang J. Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks. Tsinghua Sci Technol. 2021;26(5):646–63. doi:10.26599/TST.2021.9010007.
16. Liang CY, Liu S, Chung EY, Gaudiot JL. An energy and performance efficient dvfs scheme for irregular parallel divide-and-conquer algorithms on the AMD scc. IEEE Comput Archit Lett. 2013;13(1):13–6.
17. Alam M, Haidri RA, Shahid M. Resource-aware load balancing model for batch of tasks (BoT) with best fit migration policy on heterogeneous distributed computing systems. Int J Pervasive Comput Commun. 2020;16(2):113–41. doi:10.1108/IJPCC-10-2019-0081.
18. Baital K, Chakrabarti A. Dynamic scheduling of real-time tasks in heterogeneous multicore systems. IEEE Embedd Syst Lett. 2018;11(1):29–32. doi:10.1109/LES.2018.2846666.
19. Huang K, Wang K, Zheng D, Jiang X, Zhang X, Yan R, et al. Expected energy optimization for real-time multiprocessor socs running periodic tasks with uncertain execution time. IEEE Trans Sustain Comput. 2018;6(3):398–411. doi:10.1109/TSUSC.2018.2853621.
20. Haris J, Shafique M, Henkel J, Parameswaran S. System-level application-aware dynamic power management in adaptive pipelined MPSoCs for multimedia. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD); 2011; Washington, DC, USA. p. 616–23.
21. James B, Tenentes V, AlHashimi BM, Merrett GV. Online tuning of dynamic power management for efficient execution of interactive workloads. In: IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED); 2017; Boston, MA, USA. p. 1–6.
22. Bogdan P, Marculescu R, Jain S. Dynamic power management for multidomain system-on-chip platforms: an optimal control approach. ACM Trans Design Autom Electron Syst. 2013;18:1–20. doi:10.1145/2504904.
23. Khan H, Din IU, Ali A, Husain M. An optimal DPM based energy-aware task scheduling for performance enhancement in embedded MPSoC. Comput Mater Contin. 2023;74(1):2097–113. doi:10.32604/cmc.2023.032999.
24. Zhang X, Zhang W, Sun W, Wu H, Song A, Jha SK. A real-time cutting model based on finite element and order reduction. Comput Syst Sci Eng. 2022;43(1):1–15. doi:10.32604/csse.2022.024950.
25. Kivilcim CA, Rosing TS, Mihic K, Leblebici Y. Analysis and optimization of MPSoC reliability. J Low Power Electron. 2006;2(1):56–69. doi:10.1166/jolpe.2006.007.
26. Richard U, Deplanche AM, Trinquet Y. Storm a simulation tool for real-time multiprocessor scheduling evaluation. In: IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA); 2010; Bilbao, Spain. p. 1–8.
27. Wagner P, Strodthoff N. PTB-XL, a large publicly available electrocardiography dataset. Sci Data. 2020;7(1):1–15. doi:10.1038/s41597-020-0495-6.