DYNAMIC POWER MANAGEMENT TECHNIQUE BASED ON EFFICIENT SCHEDULING WITH TASK MIGRATION POLICY FOR MULTIPROCESSOR SYSTEM ON CHIP (MPSoC)

SUPERIOR UNIVERSITY

Ph.D. ELECTRICAL ENGINEERING

By:

HAMAYUN KHAN

ROLL NO: PHEE-F18-003

Session: 2018-21

DEPARTMENT OF ELECTRICAL ENGINEERING

SUPERIOR UNIVERSITY

Lahore, Pakistan

DYNAMIC POWER MANAGEMENT TECHNIQUE BASED ON EFFICIENT SCHEDULING WITH TASK MIGRATION POLICY FOR MULTIPROCESSOR SYSTEM ON CHIP (MPSoC)

SUPERIOR UNIVERSITY

Ph.D. ELECTRICAL ENGINEERING

By:

HAMAYUN KHAN

ROLL NO: PHEE-F18-003

Session: 2018-21

SUPERVISOR:

DR. IRFAN UD DIN

DEPARTMENT OF ELECTRICAL ENGINEERING

SUPERIOR UNIVERSITY

Lahore, Pakistan

Copyright © 2022 by Author

Hamayun Khan

Dated: Dec 15, 2022

DEDICATION

To our last prophet Muhammad (Peace be upon him), my great hero and messenger of Allah Almighty. One of the most influential people, that humanity has ever witnessed.

I dedicate the thesis to my beloved father Shaukatullah Khan, my Supervisor, my parents and wife.

RESEARCH COMPLETION CERTIFICATE

It is certified that the research work in this dissertation titled **"Dynamic Power Management Technique Based On Efficient Scheduling With Task Migration Policy For Multiprocessor System On Chip (MPSoC)"** has been investigated and carried out by **Mr. Hamayun Khan** Roll No. PHEE-F18-003 under my supervision. Therefore, the undersigned hereby certify that they have read and recommended the dissertation entitled for the degree of **Philosophy in Electrical Engineering**.

Name of the evaluator University/Institute Name (Foreign Evaluator 1)	Name of the evaluator University/Institute Name (Foreign Evaluator 2)

Dr. Irfan ud din
(Thesis Supervisor)

Dr. Muhammad Rehan Usman Thesis Committee Member	Dr. Saif Ur Rehman Thesis Committee Member

Dr. Mustafa Shakir
Chairman, Electrical Engineering Department

AUTHOR'S DECLARATION

I, **Hamayun Khan** Roll No. **PHEE-F18-003** student of Department of Electrical Engineering, Superior University, Lahore in the subject of **Ph.D. Electrical Engineering** Session 2018-21, hereby declare that the matter printed in the dissertation as **"Dynamic Power Management Technique Based On Efficient Scheduling With Task Migration Policy For Multiprocessor System On Chip (MPSoC)"** is my research work. The text and results mentioned in this dissertation have not been printed, published, or submitted in any form at any national or international organization. I hereby certify that this research does not involve any plagiarized material or results that have been published by another person. My colleagues and friends assisted me while carrying out this research; I identified their contributions as well. If it contains any plagiarized information, I bear full responsibility.

Signature:

Dated: Dec 15, 2022

_____

Hamayun Khan

PLAGIARISM UNDERTAKING

I solemnly state that the research work illustrated in this dissertation titled **"Dynamic Power Management Technique Based On Efficient Scheduling with Task Migration Policy For Multiprocessor System On Chip (MPSoC)"** is my own developed and proposed work without any remarkable contribution of any other person.

I also recognize the HEC and Superior University, Lahore zero-tolerance policy towards plagiarism. Therefore, I as an author of the above-titled dissertation declare that this research does not involve any plagiarized material or results that have been published by another person, and literature with reference is properly cited in the bibliography section.

I bear full responsibility that if I am found guilty of any kind of plagiarism in the above-titled dissertation even after the award of my degree, the University reserves the right to revoke my degree at any stage. Furthermore, HEC and the University have the right to declare blacklisted at any forum.

Hamayun Khan

Dated: Dec 15, 2022

LIST OF PUBLICATIONS

It is certified that the following publication(s) have been made of the research work that has been carried out for this thesis:

Journal Articles:

1. **Hamayun Khan, Irfan Ud din**, Arshad Ali and Sami Alshmrany, **"Energy-Efficient Scheduling Based on Task Migration Policy Using DPM for Homogeneous MPSoCs "**,Computers, Materials & Continua 2022, Vol. 74, Issue 01. DOI: 10.32604/cmc.2022. 032999 (IF: 3.86) (Category: W).
2. **Hamayun Khan, Irfan Ud din**, Arshad Ali and Mohammad Husain, **"An Optimal DPM Based Energy-Aware Task Scheduling for Performance Enhancement In Embedded MPSoC"**, Computers, Materials & Continua 2022, Vol. 74, Issue 01. DOI: 10.32604/cmc.2022.032999 (IF: 3.86) (Category: W).

Hamayun Khan

Dated: Dec 15, 2022

ACKNOWLEDGEMENTS

In the name of Allah Almighty, the Most Beneficent, the Most Merciful. First and foremost, I would like to thank Almighty Allah for making all this possible for me and for bestowing upon me uncountable blessings. I express my sincere gratitude to my parents and supervisor for their invaluable leadership and inspiration throughout my educational tenure at Superior University.

I am extremely grateful and would like to express my sincere gratitude to the sincere person my supervisor **Dr. Irfan ud din**, for his continuous encouragement and motivation towards pursuing a Doctor of Philosophy Degree in the field of Electrical Engineering.

I would like to acknowledge all faculty and staff members of Superior University, Lahore for the provision of a supportive and working environment.

In the end, I am very thankful to my parents, who have always inspired me with their inspiration and praise. They also provided me a carefree environment, to help me maintain focus on my study with complete concentration. Even though they do not know what research writing is and what is my area of research. My father is the most enthusiastic to support all the decisions which I made. May Allah Almighty give them good health and keep them safe. Finally, to every person who directly or indirectly contributed to this work, their kindness means a lot to me. Thank you all very much.

Hamayun Khan

Dated: Dec 15, 2022

ABSTRACT

Energy optimization is one of the critical design concerns for multiprocessor systems. With the advancement of technology the performance management of the central processing unit (CPU) is changing. Power densities and thermal effects are quickly increasing due to shrinking of chip size and increase in deployment of per inch on-chip transistors causing to increase in energy dissipation and is a serious design issue in multi-core embedded technologies. Increasing the life span and efficiency by reducing energy utilization has become a critical chip design challenge for multiprocessor systems on chips (MPSoCs). This research addresses the issue of energy-aware task scheduling & high energy utilization problem in (MPSoC) by introducing an optimal energy-aware earliest deadline first scheduling (EA-EDF) based on the dynamic power management (DPM) technique using task migration to enhance the performance and efficiency in multiprocessor system-on-chip while reducing the energy consumption using DPM low power mode. Meeting the time constraints of application dissipates higher energy for high performance while on other hand switching MPSoC to a lower state for reducing the consumption of energy that causes a delay in execution and performance degradation. In our proposed technique schedulers allocate resources to the task so that their deadlines are guaranteed. Efficient task scheduling and selection of core for migration of task by considering (DPM) policy prevents the system from reaching its maximum energy utilization. DPM mechanism reduces the consumption of energy by switching the CPU state according to the utilization factor. Due to high task execution the utilization factor (u_i) on-chip reaches the maximum allowable threshold that dissipates more energy. The proposed approach migrates such tasks to the core that are least used or consumes less power and energy by distributing the load on other cores to lower the energy and optimize the duration of idle and sleep times across multiple CPUs. The performance of the proposed EA-EDF algorithm using wide set of experiments, reported excellent significant improved results when compared to other current techniques, the efficacy of the proposed methodology reduces the energy consumption by 4.3%,4.7%, 5.3%,5.9% and 6.3% on (u_i) of 6%, 36 % & 50 % at various operating frequencies (624 MHz, 520 MHz, 416,312 MHz,208 MHz,104 MHz) when particularly in comparison to other conventional energy-aware methods for MPSoCs. Tasks are running and accurately scheduled to make an energy-efficient processor system model by controlling and managing the effects of high utilization on-chip and optimizing the overall energy consumption of MPSoC.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

AET	Actual Execution Time
BCET	Best Case Execution Time
BF	Best Fit
DVFS	Dynamic Voltage and Frequency Scaling
DPM	Dynamic Power Management
DBF	Demand-Bound Function
DTM	Dynamic Thermal Management
DVS	Dynamic Voltage Scaling
DP	Dynamic Priority
EDF	Earliest Deadline First
FJP	Fixed Job-Priority
FTP	Fixed Task-Priority
FFD	First-Fit Decreasing
ILP	Integer Linear Programming
MPSoC	Multiprocessor System-on-Chip
NL-P	Nonlinear Programming
RF	Random Fit
RTOS	Real-Time Operating System.
RT	Real Time
SoC	System-on-Chip
SMP	Symmetric Shared-Memory Multiprocessor
UMA	Uniform Memory Access
WCET	Worst-Case Execution Time

LIST OF SYMBOLS

Symbols	Meaning/Description
τ	Task Set
τ_i	i_{th} task of an application
D_a	The predefined deadline of τ_i
u_i	Utilization factor
P	The application period
$WCET\ (\tau_i)$	Worst case execution time of τ_i
E_i	The execution time of τ_i
$pred(\tau_i)$	The predecessors of τ_i
V_{ddmin}	The minimum supply voltage of a processor
V_{ddi}	The supply voltage of τ_i
E_{dyn}	Dynamic energy consumption of MPSoC per cycle
C_{eff}	The capacitance of a processor due to switching
V_{th1}	The threshold voltage
L_a	The total no of logic gates
P	A set of processors
P_K	The k_{th} processor
D_{ji}	The successor-tree-consistent deadline of τ_i
P_{static}	Static power
S_i	Starting time of τ_i
C_{ci}	The total no of clock cycles of τ_i
C_i	Count time for core selection
$succ\ (\tau_i)$	All successors of τ_i
V_{ddmax}	Maximum Supply Voltage of *P*
f_i	The operating frequency of τ_i
$\tau ready$	Ready task set
∂	Configurations of Core selection
E_{tot}	The total energy consumption of processor
L_g	Number of logic gates in a MPSoC

V_{bs}	The body bias voltage of the MPSoC
I_j	The body junction leakage Current
I_{rbj}	Reverse bias junction current
∂_a	Densities of MPSoC
I_{st}	Sub-threshold leakage
$E_{Per-Cycle}$	Total energy consumption per cycle
$E_{running}$	Running state per cycle
$Pi(s)$	Power consumption function
Ci	CPU-cycles
T_{be}	Breakeven time
N_{cs}	Number of circuit transitions
C_{load}	Load capacitance
S_p	Schedule of processor
p	Denotes processor
$m(p)$	Total no of CPU in p
$s_{p(i)}$	Speed of i_{th} processor
$s_{(p)}$	The average processing power of the processor
$s_{i(p)}$	The average processing power of the i_{th} processor
$P_{running}\ (\alpha)$	Represents the processor executing τ denoted by α
$P_{idle}\ (\Upsilon)$	Represents the m processor in idle state denoted by Υ

CHAPTER 1
INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 Introduction

Multiprocessor Systems-on-Chips (MPSoCs) are increasingly in demand and used in multicore embedded systems. MPSoC consists of multiple processors and all the components are on the same chip consume low energy and required less power because of the tightly integrated architecture [1]. 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. In recent years the demand for the (MPSoC) has grown exponentially from consumer devices (e.g., Mobile desktops and notebooks) to embedded sectors with high-performance computing systems [2]. MPSoC is widely deployed in high-performance systems for real-time response and specific embedded systems based on ARM Cortex-A7, A15 and high-performance MARVEL INTEL PXA-270, 250 multi-processor [3]. The authors in [4] introduced a technique that reduces resistance and energy because lack of concentration can affect the reliability and life span of the chip as well as overall systems performance. Due to the high processing of tasks temperature on the chip increases. Various mechanisms are used to reduce the thermal effects due to high heat and decrease the performance of the system because high heat causes the chip to be damaged [5]. A central processing unit is working as the main processing unit for performing the instructions read and write operation. CPU unit is placed in all the modern embedded systems [6]. The processing unit needs to be updated with time using integer linear programming (ILP) if the CPU has a higher processing speed it can manage critical tasks efficiently at short intervals of time. Advancement in the processing unit makes our system run heavy tasks but it can have some issues e.g. dimension, cost, energy, power utilization, performance, reliability and processing speed. Switching of the task is the major issue with the evolvement of the processor [7]. Introduced simulated annealing based ($LPPWU_{Sa}$) optimization strategies for reducing the energy and solving the system-level low power design problems [8].

The authors in [9] proposed dynamic thermal management (DTM) based energy optimization technique for the MPSoC platform with dynamic voltage and frequency scaling (DVFS) enabled homogeneous processors. These are one of the most reliable techniques to reduce and stabilize the temperature of the multi-core system. In a multi-

core system, an exponential decrease in the temperature also reduces power utilization. The authors in [10] introduced efficient homogeneous multiprocessing often known as symmetric multiprocessing (SMP) for identical processors that share a single main memory for process execution. The authors in [11] introduced dynamic thermal management (DTM) based technique that efficiently manages the thermal responses of a processing system. Many techniques are combined to manage temperature and thermal responses including DVFS, DPM and dynamic voltage scaling (DVS). These techniques are very useful but they cause some reliability and performance issues. They are mostly used to resolve on-chip power dissipation problems.

The authors in [12] introduce an efficient task migration policy that moves an executing task from one host CPU in a distributed architecture to another system to another. The selection of task and movement to the host CPU for a new task and the creation of the task on that host increases the performance, reliability and processing speed. The lack of task migration on time can affect the overall system performance. The authors in [13] The temperature on chip increases for which various mechanisms are used to reduce the thermal effects due to high heat and increase the performance of the system because high heat causes the chip to be damaged.

There are a few more techniques recently introduced for managing voltage, frequency and power i.e. (DTM, DVFS, DPM) and integrated DVS. The effect of thermal cycles and temperature gradients are not considered in integrated dynamic voltage and frequency scaling that increases the reliability and performance of the system. Fetch toggling and clock gating is a hardware technique used to increase the chip temperature because these hardware techniques don’t consider appropriate information [14]. Introduces a technique that improves the performance and reliability of a multi-core processor by checking the impact of power consumption of a multi-core processor and dynamically checking the frequency and voltage. This approach is also used to predict the thermal variation on each core [15]. The proposed method gives a 54.3% improvement in results and 61% improvement in energy delay as compared to the conventional multi-processor using the best speed fit the earliest deadline first EDF technique that chooses the suitable processor when the task is allocated for execution and the proposed technique doesn’t consider the fastest core.

The performance and reliability of the chip are improved than that of the global earliest deadline first algorithm that considers the priority of the task while executing. Migration

of tasks can occur at any time with different speed processors and tasks were assigned based on deadlines those tasks that can have an early deadline can have higher priority [16]. In [17] proposed a last-level cache (LLC) technique is used to reduce the heat that produces high leakage power. This technique is used to increase the performance and reliability of the system. The effect due to high temperature on chip causes various thermal issues due to high heat on chip the cooling cost also increases. When the temperature of the chip reduces to 4℃ a simultaneous 52% maximum savings in cache leakage occurs.

In [18] introduce a dynamic voltage and frequency scaling DVFS-based technique for a single level CPU that comprises scaling of the large unit to scaling of any single logic block on critical mode. The DVFS technique reduces $P_{dynamic}$ as well as P_{static} but it's also dependent on the supplied voltage V_{dd} as well. The authors in [19] measure the total energy consumption of the MPSoC using the DVFS technique as shown below in Equations 1.1 and 1.2 respectively.

$$E_{total} = E_{sen} + E_{proc} + E_{Comm} \tag{1.1}$$

Reducing the energy among each CPU core also reduces the communication energy represented as n_a bits are given as follows.

$$E_{TR}(n_a, d)(\geq \begin{cases} n_a.E_{ss} + n_a.e_{ffs}.(d^2) & d \leq d_o \\ n_a.E_{ss} + n_a.e_{ffp} * (d^4) & d \geq d_o \end{cases} \tag{1.2}$$

Whereas E_{ss}represents the energy dissipation per bit in the transmitter circuit, e_{ffp} and e_{ffs} are the amplification energy. In [20] proposed an advanced DVFS technique that is used to improve performance and consider the imbalance issue of workload on various cores of a processor and dynamically set the workload on the core equally for this an irregular parallel divide-and-conquer algorithms is used that reduces 31% of energy consumption at 400MHz [21]. A decrease in the chip size of a multi-core processor certainly increases the electronic element transistor on a chip more rapidly than before. The chip can require more energy due to which a gradual increase in power density is observed that affects the reliability of a multi-core processor [22]. High power density also causes many thermal issues as an increase in temperature these thermal issues can disturb the performance of the system. It's better to decrease the temperature of the

processor by removing such a task that produces heat. The chip can have a stable temperature and reduces the cost required for the cooling of the chip. While the cost of cooling can be gradually increasing when hot tasks are executed and intense heat produces on-chip [23]. The authors in [24] proposed that the power usage in the chip can be efficiently managed by using the power management technique PMT. This technique is used to reduce the power consumption of a chip because all the cores that are not running are turned off due to which the performance of the system increases by decreasing the overall chip temperature the power consumption is also decreased. In [25] elaborates the effects of gradual increase in the system on chip SoC transistors that increases the dissipation of power and energy. As Moore's empirical law states that the transistors on the chip doubles after every eight months as shown in Figure 1.1 the transistors on the chip doubles after every eight months.

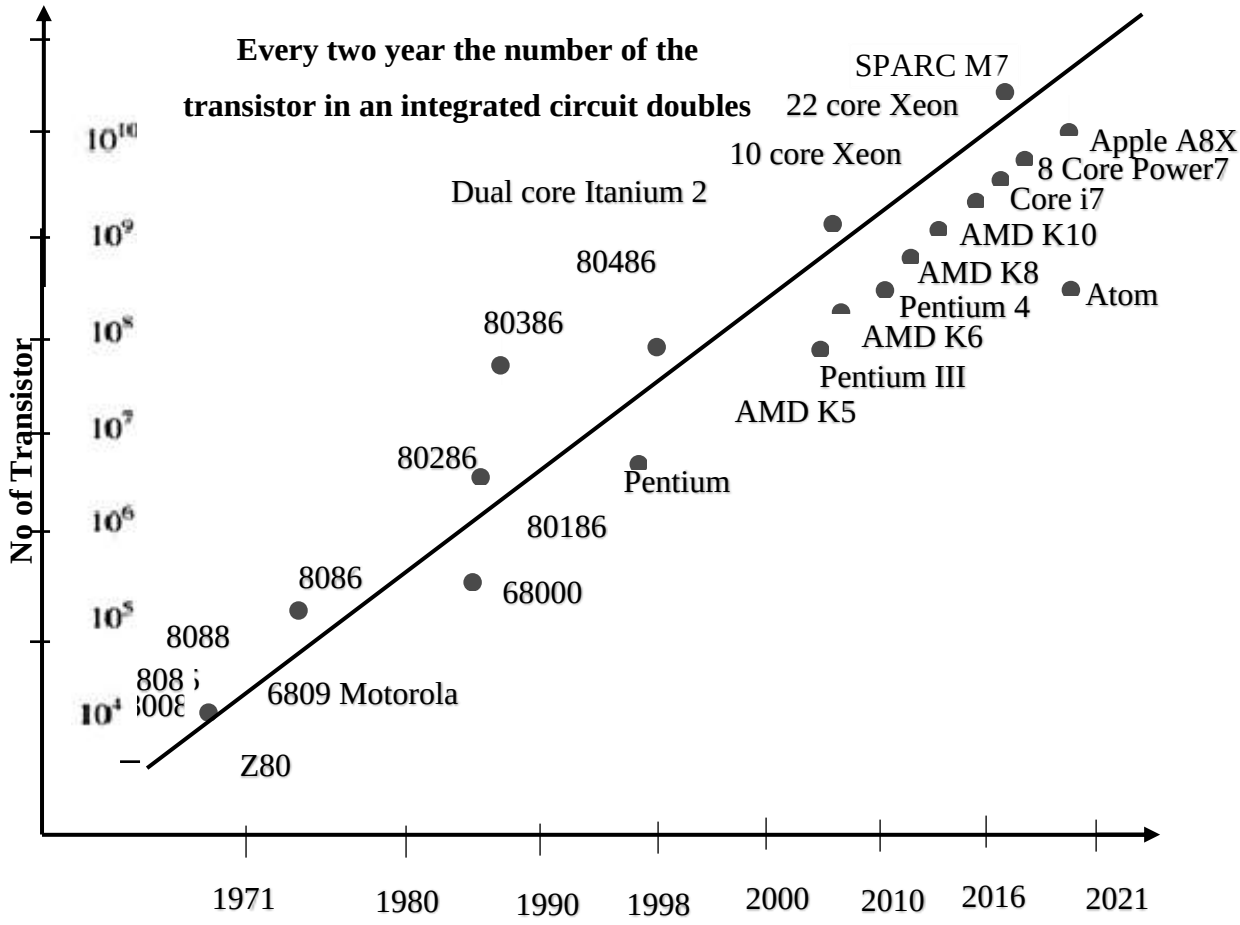


Figure 1. 1 *Rapid increase stated in Moore's empirical law of transistors on SoC* [26].

That increases the on-chip power densities for such a reason DPM and dynamic frequency scaling (DFS) techniques are introduced to reduce the transition rate and energy dissipation. The authors in [27] introduced a core selection and configuration

method when the transition occurs from a high power state to a low power state these transitions can disturb the processor performance.

1.2 Background

Embedded systems are designed to combine hardware and software. The software can insert into hardware that is used for a specific function. They are in different shapes and dimensions. It has various applications on a larger scale in networking and nuclear powerhouses to small MP3 and MP4 player printers, automobiles, cameras mobile handsets [28]. A power (PMU) and energy management unit (EMU) is integrated into a system together that is widely used for managing the DVFS & DPM-based MPSoC device. It is located on a common IC. Figure 1.2 illustrates a block diagram of an MPSoC along with an exemplary PMU/EMU based on the DPM technique.

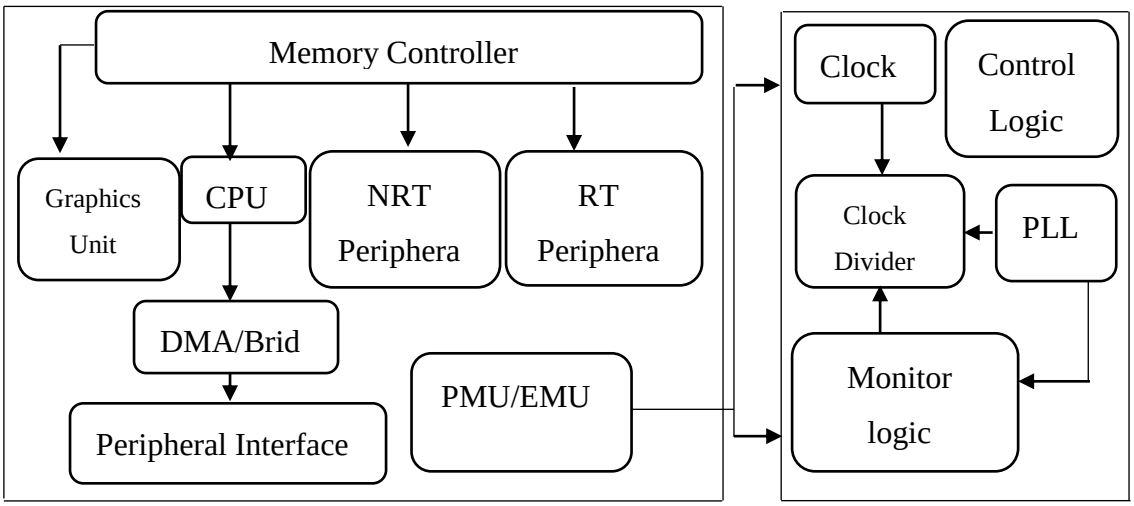


Figure 1. 2 *Block diagram of PMU/EMU based Embedded System* [28]

Software is already built in for all applications in embedded systems. Applications can be operated and run without human interference. The most frequently used embedded systems are laptops and cellular phones. These devices have less memory and smaller data storage mechanism. Embedded systems often have limited memory, due to improvements in the internet of things IoTs, embedded systems become more difficult [29]. Now a day's systems are getting advanced and they are characterized as they have real-time necessities to operate efficiently. The authors in [30] introduce a genetic algorithm (GA) for task migration that is an important concern in modern MPSoC-based embedded systems to reduce the consumption of energy. In such embedded machines the performance of the system does not consider those results that come at the end of the complete simulation of a task in a multi-core system but considers the duration during

which the required results can be achieved such a kind of system is generally recognized as a real-time system. The authors in [31] introduce the various layers of embedded systems These systems are usually linked to such applications that need to have a smooth operation for all the applications that required reliability and cannot afford any erroneous results in such real-time systems all the deadlines meet at an accurate time and no chances for the system to miss its deadline.

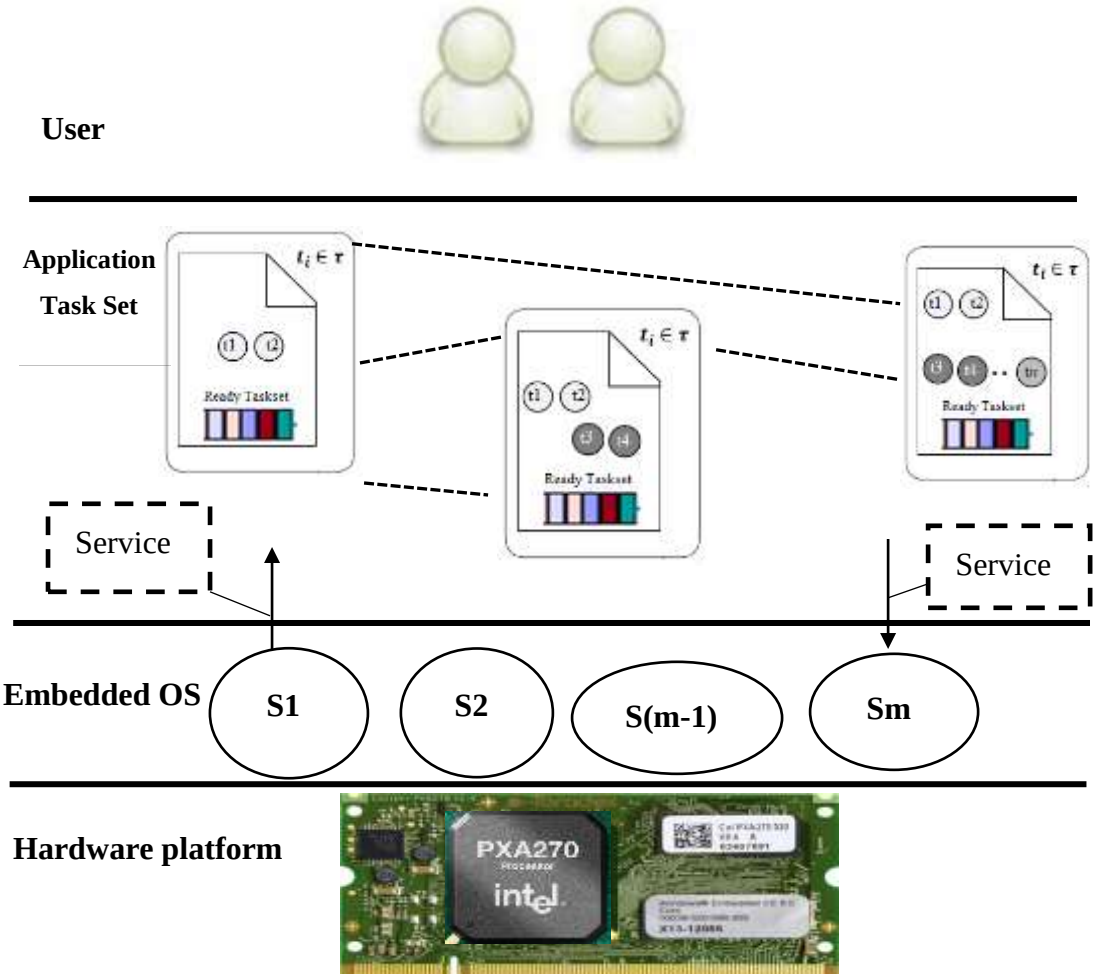


Figure 1. 3 *Various layers of an embedded system* [31]

If a deadline is missed a failure as a response can occur. These devices are called hard real-time systems a very common example on a larger scale for hard real-time systems are avionics systems, textile industries, nuclear power plants, and also those devices that operate with wire systems established by present advanced automobiles [32]. In systems in which the real-time limit exists, the system can function whether a multi-core system missed the deadlines or either released late. Such a system is recognized as a soft real-time embedded system. These embedded devices are safe as compared to hard ones. If the time limit for each task is missed but yet the system is functional such systems are

known as soft real-time systems [33]. The performance of the system can be slightly disturbed and can affect the total performance of the system. Such systems can be used in the networking of cellular systems. Once the deadline of the task arrives firm real-time systems can afford a delay of a few seconds like a server of the network [34]. The below section elaborates the real-time system.

1.3 Real-time systems

In a real-time system instance $\boldsymbol{i} = \{\boldsymbol{t_1}, \boldsymbol{t_2}, \ldots, \boldsymbol{t_n}\}$ contains the time-constrained jobs. Each task $\boldsymbol{t_i} \in \boldsymbol{i}$ consists of $(\boldsymbol{r_i}, \boldsymbol{c_i}, \boldsymbol{d_i}, \boldsymbol{p_i}, \boldsymbol{p_r})$, where the release time of a task is denoted as $\boldsymbol{r_i}$, while $\boldsymbol{c_i}$ is the worst-case execution time, and the deadline is denoted as $\boldsymbol{d_i}$. Period and priority are represented as $(\boldsymbol{p_i}, \boldsymbol{p_r})$. A task at the instance $(\boldsymbol{t_1})$ activate at *t* if $\boldsymbol{t} \geq \boldsymbol{r_i}$ and $\boldsymbol{t}$ cannot be executed at $(\boldsymbol{c_i})$ units by time$(\boldsymbol{t})$. A usual real-time system can behave as a hybrid and sometimes it behaves as a hard real-time for which all the deadlines can meet according to their parameters [35]. Hard, soft, and firm are those real-time systems that are commonly used everywhere and are linked with a conventional multi-core system. These are further categorized into few other types like the interactive real-time that gives a response to a user while the application is in running mode these types of embedded devices depend on time so the user can expect the response of the system well on time [36]. Users can have a better system speed and have a continuous process and have very less chances of missing a deadline such systems have almost the same characteristics as soft real-time systems [37]. Devices that are in the real-time embedded system category have a larger number of common characteristics. Real-time systems have characteristics like embedded while embedded systems usually have characteristics like real-time on a larger level [38]. In [39] introduced a type of event in which the release time assigned to all the tasks of an application is either the same or random depending upon the application. Figure 1.4 illustrates a real-time embedded system.

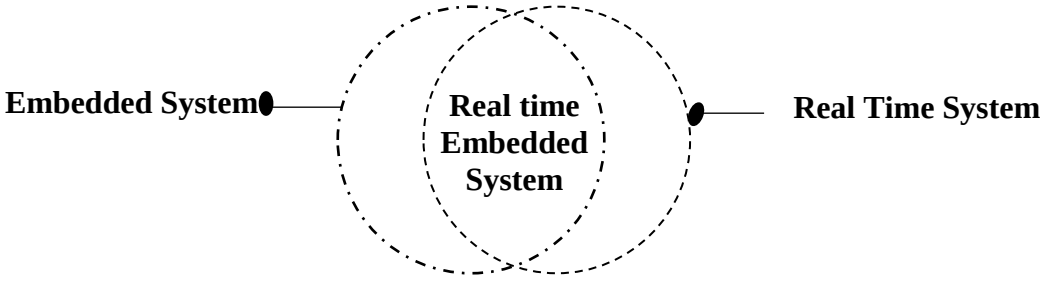


Figure 1. 4 *Illustrates a real-time embedded system* [40]

In [41] introduce real-time embedded system RTES are systems for which it is necessary to execute the set of jobs allocated to it firmly within the pre-set limit restrictions. It guarantees that all time-critical tasks are handled within time called a real-time system with hard deadlines. These tasks trust deeply on computational analysis and data plays an important role in the achievement of the specific task to avoid scheduling issues all the resources that are distributed try to meet the time constraints for the task not to miss any deadlines [42]. Figure 1.5 illustrates a real-time system that works with synchronous and asynchronous events.

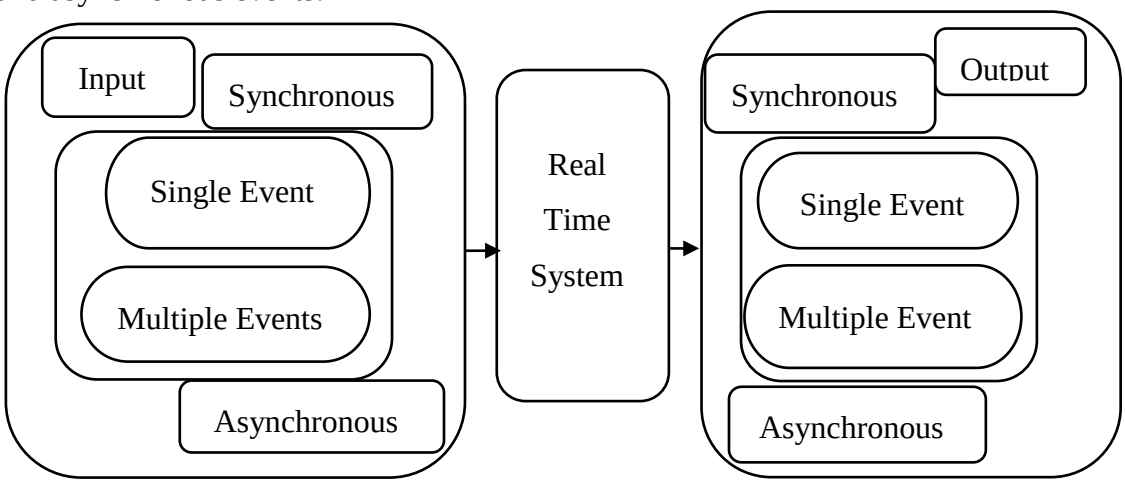


Figure 1. 5 *Behavior of real-time systems with same & random release time* [43]

1.4 Energy Aware Multiprocessor System-on-Chips (MPSoCs)

Today's energy-efficient and high-performance systems are developed using multiprocessor systems. A silicon chip that contains many independent and connected processors is known as MPSoC. These processors work together and share information during the execution of programs [44]. An MPSoC consist of a single chip system that combines all the functions of an electronic system, including memory, specialized logic, instruction-set processors, and digital signal processing operations. I/O units [45]. The MPSoC has given the area of embedded systems a new direction including contemporary MPSoC designs with multi-core capabilities, Graphics Processing Units (GPU), (3G, 4G, WiFi, and 4G LTE) [46]. Embedded multiprocessors are deployed in applications like vehicles, airplanes, robots, and High-performance computing systems [47]. Automation and manufacturing and unmanned aerial vehicle (UAVs). IOTs and nuclear power plants [48,49] as shown in Figure 1.6 the increase in demands for more complex applications has resulted in modern embedded systems being driven by multiprocessor architectures.

Moreover, due to the increase in demand multiprocessor architectures such as (MPSoC) have received attention due to their high performance and low power requirements [50].

Figure 1. 6 *Marvel Intel PXA270 MPSoC in embedded system applications* [50]

Based on MPSoC communication, designs and the processors inside each MPSoC system it's divided into three categories as shown in Figure 1.7 [51].

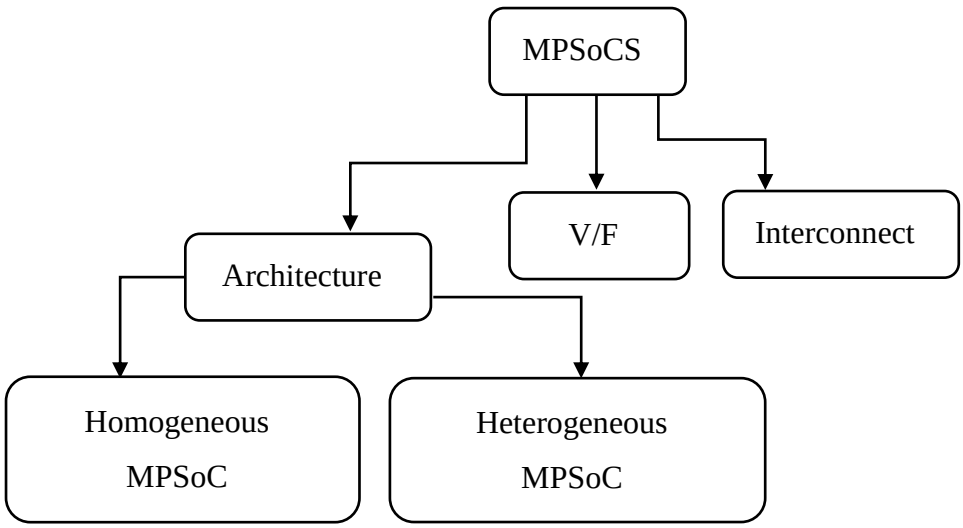


Figure 1. 7 *Multi-Processor System on Chip Classification*

1.4.1 MPSoC Architecture

MPSoCs architecture can be divided into two general groups homogeneous and heterogeneous MPSoCs.

1.4.1.1 Homogeneous MPSoCs

Homogeneous processors are those multi-core architectures in which several cores with the same characteristics are integrated on a single chip e.g. processors with the same characteristics of multi-cores use two or four core processors [52, 53]. There are a few issues in homogeneous processors. Wastage of resources has occurred when a task is divided parallel that reducing the performance and reliability of the system. The authors in [54, 55] introduced symmetric multiprocessing based on homogeneous MPSoC in which the same instruction set architecture (ISA) is used. The authors in [56] introduce various homogeneous MPSoCs that are efficient platforms for computing applications where performance and the communication ratio are higher than these homogeneous. MPSoC includes Intel PXA270 PXA250, Cortex-A53 processors and TILE-Mx100TM Cortex-A53 processors.

1.4.1.2 Symmetric Multi-Processing

In symmetric multiprocessing real load sharing can be occurred when all the processors in a multi-core system are the same. There are two ways for load sharing between two processors. Every individual processor can have a separate queue for task set and processes follow and enter into the nearest direct ready queue. For all the processors there is a single ready queue for the process [57]. The authors in [58] illustrate that all the processors are sharing the same main memory and are peers to one another all the processors are performing their specific tasks as shown in Figure 1.8.

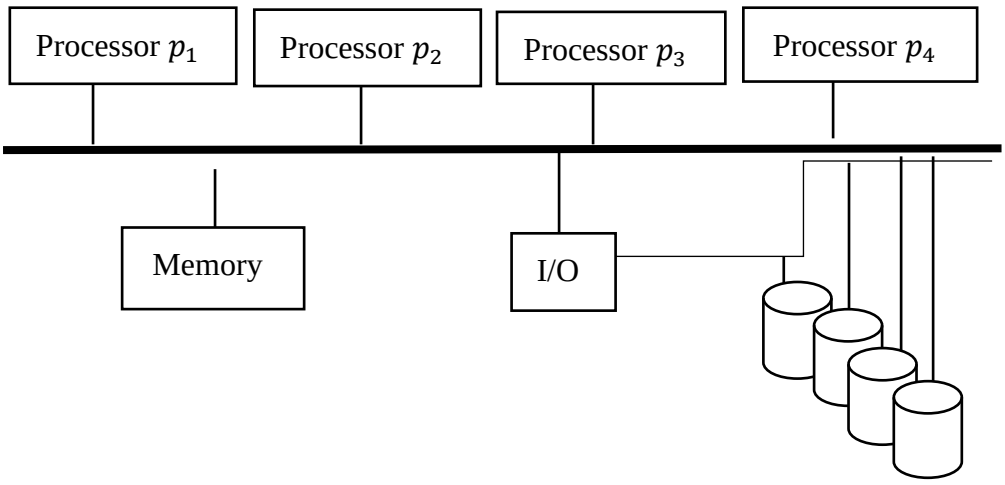


Figure 1. 8 *Shared memory-based symmetric multi-processing* [58]

The next process can be executed from a similar spot. Every Individual processor is a peer to another processor and a single processor can execute all the periodic tasks appearing one after other in an operating environment.

1.4.1.3 Heterogeneous MPSoCs

The authors in [59] proposed energy-efficient multi-core architectures that use a different kinds of cores in a processor. The performance, reliability and throughput of the system increase due to low power dissipation in heterogeneous MPSoCs. Single independent instruction or either task can be executed at a time. A Dual-core can execute two independent instructions and a quad-core can execute four autonomous programs at a time and an octa-core processor can execute eight independent instructions/tasks at a time. This MPSoC contains multiple processing elements. It can either works as functional or performance asymmetric multiprocessing [60].

1.4.1.4 Asymmetric Multi-Processing

In asymmetric multiprocessing, there is a specialized way for the process of a task set in a multi-core system. There is a single processor for a system data structure. For the input/output of the system, there is a single processor in the system [61]. Asymmetric multiprocessing is a very easy way the sharing load using the processor master and slave technique. Slave processors can have specified jobs to perform or either they are waiting for the instructions coming from a master processor to perform the specific task allocated by the master processor [62]. In [63] introduces design constraints of an MPSoC when memory is not shared and every processor has its specified task and memory resource allocated by the master processor as shown in Figure 1.9.

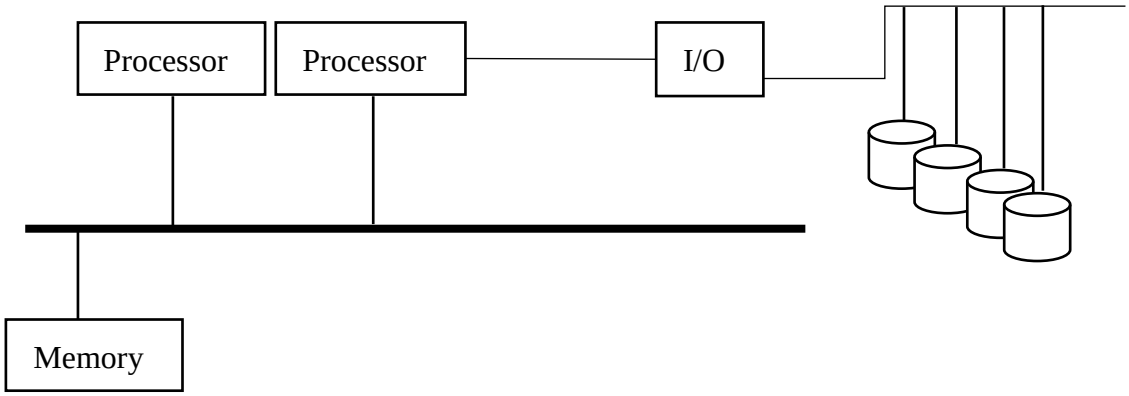


Figure 1. 9 *Asymmetric multi-processing when Memory is not shared* [63]

1.4.2 Interconnect

The authors in [64] introduce an MPSoC based on the interconnect technique that includes the arrangement used for inter-processor CPU communication. Inter-processor

communication is used for high performance and energy efficiency by reducing communication congestion.

1.4.3 Voltage Frequency Islands-based MPSoCs

In [65] introduces voltage frequency island (VFI) to network-on-chip (NoC) is an important element in chip multiprocessors (CMPs) that support communication between various cores. In VFI the tiles are partitioned into islands that are optimized with their frequency, threshold and supply voltage. The authors in [66] proposed a VFI-based mechanism for reducing the power consumption and enhanced complexity of the MPSoC that increases when the number of VFI's is increased.

1.5 Dissipation in Multiprocessor system on chip (MPSoC)

The dissipation of energy refers to the discharge of energy from MPSoC based embedded system. The dissipation of energy occurs when the embedded system utilizes energy for its operation and functioning. The most demanding and widespread electronic digital circuit technology is complementary metal-oxide semiconductor CMOS whose dissipation of peak power occurs when the transistor changes its state. Modern MPSoCs are based on CMOS chips. In UMA switching frequency regulates how often the switches occur. Double-edge-triggered flip-flops can be utilized to minimize dynamic power for MPSOC technology. An embedded device's power consumption is categorized as $P_{dynamic}$ and P_{static} [67].

The authors in [68] introduce a static power mechanism in which the dissipation of power occurs when the circuit is not changing states due to leakage current. The short circuit power is utilized when both positive channel metal-oxide-semiconductor (PMOS) and negative channel metal-oxide-semiconductor (NMOS) are switched on for a short period unless the path between supply voltage is directed with the ground. Various low-power states are introduced to address such issues, by reducing the dissipation of power when the CPU is not executing any task [69].

The authors in [70] calculate the accumulative power of a CPU that is the sum of the $P_{dynamic}$ because of switching the P_{static} occurs due to leakage of power. The primary factor in CMOS circuits is dynamic power dissipation that occurs because of transistor switching. $P_{dynamic}$ can be calculated using below mentioned Equations.

$$P_{dynamic} = C_{load}.N_{cs}.V_{dd}.f \quad (1.3)$$

$$f = R.(V_{dd} - V_t)/.V_{dd.}{}^2 f \quad (1.4)$$

$$P_{total} = (C_{eff}\, V_{dd.}{}^2 f + L_a(V_{dd.}R_{3.}) \quad (1.5)$$

Load capacitance is denoted as C_{load}, N_{cs} is the number of circuit transitions while supplied voltage is denoted as *V* which is equivalent to switching voltage. *f* is the clock frequency while V_t represents the threshold voltage. R is a constant and a reduction in supply voltage is represented as V_{dd} as shown in Equation 1.4. L_a is the total number of logic gates in the CMOS circuit. The total power of the CMOS circuit that plays an important role in modern multiprocessors as illustrated in Equation 1.5. In [71] introduce a technique that measures the clock cycle of the CPU T_{cycle} represents the length of the cycle of the clock for the execution of an assigned task using a specific speed/voltage level that can be expressed in Equation 1.6.

$$T_{cycle} = L_{dep} * M_6 * / V_{dd} - V_{th} \quad (1.6)$$

Where M_6 represents the technology-dependent constant, L_{dep} shows the processor's average logic depth using the critical path. The authors in [72] proposed a technique that calculates the threshold using V_{th} in Equation 1.7.

$$V_{th} = V_{th1} - M_1, V_1 - M_2 \quad (1.7)$$

Most of the embedded computing circuits aim to give a maximum performance while using minimum power. In [73] proposed a mechanism for unmanaged thermal control as many problems occur due to increase in temperature and energy density e.g. hotspot appears due to high temperature on-chip thermal stabilization process is used to manage the power. The authors in [74] introduce that a decrease in the chip size of a multi-core processor certainly increases the number of transistors on a chip more rapidly than before. The chip can require more energy due to which a gradual increase in power density is observed that affects the reliability of a multi-core processor.

The authors in [75] introduces a mechanism that measures the higher energy dissipation and increases temperature on chip which increases the resistance eventually causing lower possible speed and hotspots that cause the permanent failure of the device as shown in Figure. 1.10

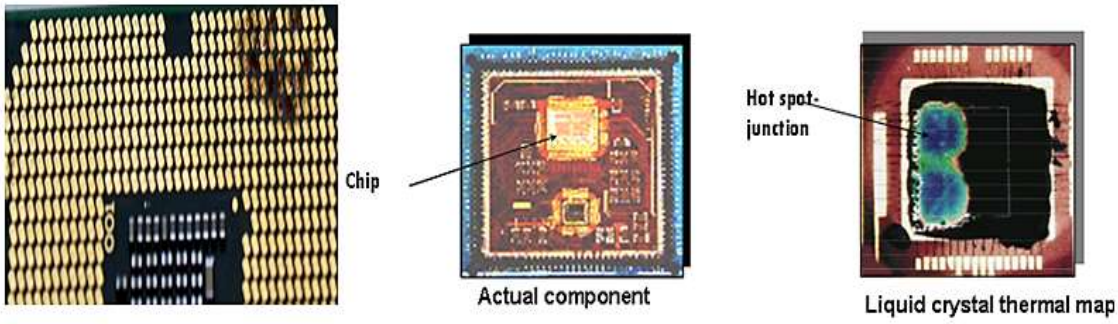


Figure 1. 10 *High utilization effects on System on Chip (SOC)* [75]

The authors in [76] introduce a technique that increases the life span of the chip by reducing the consumption of power in multiprocessors-based systems.

Current MPSoC can switch its state when not required to low-power states when the execution of the task is suspended the feature improves the performance and considers the load balancing issue in multi-cores of a processor.

1.5.1 Energy/Power in Multiprocessor system on chip

Energy and power are used alternatively but both are different from each other in physical values while power is the rate of energy consumption [77]. The energy required for MPSoC to execute a task is shown in Equation 1.8.

$$E_i = R.N.T_i \tag{1.8}$$

E_i represents the energy, R is a constant factor, N denotes the total no of the CPU clock cycle and T_i denotes the clock period. The authors in [78] measure the average power is using Equation 1.9.

$$P_i = I.V_{dd} \tag{1.9}$$

P_i represents the amount of average power, V_{dd} denotes the supply voltage and I represent the current.

The authors in [79, 80] proposed an LC-Earliest deadline first-based scheduling algorithm 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 α_k when another task with the nearest deadlines is executed that the t_i can be delayed using.

$$\sum_{i=\{1,,,,n\}/k}^{n} \frac{c_i}{\tau_i} + \frac{c_k + \alpha_k}{\tau_k} = 1 \tag{1.10}$$

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 λ_j and max time duration for the idle period is represented as α_j can be measured using Equation 1.11.

$$\sum_{i=\{1,,,,n\}/(k,j)}^{n} \frac{c_i}{\tau_i} + \frac{c_k + \alpha_k}{\tau_k} + \frac{c_j + \alpha_j}{\tau_j} = 1 \tag{1.11}$$

The techniques for low power scheduling of DPM are applied to fixed or varying task sets on both the single and multicore embedded systems [81]. In [82] introduce a DPM-based technique that calculates the breakeven time as shown in Equation 1.12 represents the parameter $\boldsymbol{B_e}$ known as the break-even time. $\boldsymbol{B_e}$ refers to the minimum duration of idle interval provided to schedule and utilizes the sleep state $\boldsymbol{\delta x}$ efficiently. $\boldsymbol{B_e}$ is the sum of the duration to complete the state transition and the duration of idle time that is required to find a way to reduce the shifting energy [83].

$$TB_e = \max(\delta x \frac{E_x - \delta x_s * P_{\sigma x}}{P_u - P_{\delta x}}) \tag{1.12}$$

The below Equation 1.13 determines the $\boldsymbol{BET_{sleep}}$ break-even time.

$$BET_{sleep} = \max(t_{pa} \frac{E_{Pa-}P_{sleep} * t_{pa}}{P_{idle} - P_{sleep}}) \tag{1.13}$$

The authors in [84] introduce a mechanism for short circuit power that is utilized when both PMOS and NMOS are on for a short period. Equation 1.14 represents the transition of the state $\boldsymbol{E_0}$ and its power dissipation in running and the idle state a $\boldsymbol{P_w}$ and $\boldsymbol{P_s}$.

$$P_w = T_{be} = E_0 + P_s * (T_{be} - T_0) \tag{1.14}$$

In Equation 1.15 (T_{be}) breakeven time measures the length of the idle state of the CPU to optimize power.

$$T_{be} = \left(\frac{E_0 - P_s * T_0}{P_a - P_s}\right) \tag{1.15}$$

During the running state of an application, a selective shut-down of the system components occur that are in the idle state increases the performance [85]. CPU starts to transition the energy required for the state transition from sleep to idle and from idle to running is represented as E_i and its power dissipation at state 1 is denoted as P_a and at state 2 its P_s. High-performance delay due to state transition that must be less than T_{be} as shown in Equation 1.16.

$$T_{be} = max\left[\left(\frac{E_i - P_s * c}{P_a - P_s}\right) * T_0\right] \quad (1.16)$$

In [86] introduce a technique for energy reduction using an operating system level is implemented to optimize power and energy mainly by switching resources. Figure 1.11 illustrates the behavior of MPSoC. On the left side, the MPSoC is running while on the other hand, the device is in an idle state. The energy consumption on both ends is equal because of the break-even time in DPM-based techniques.

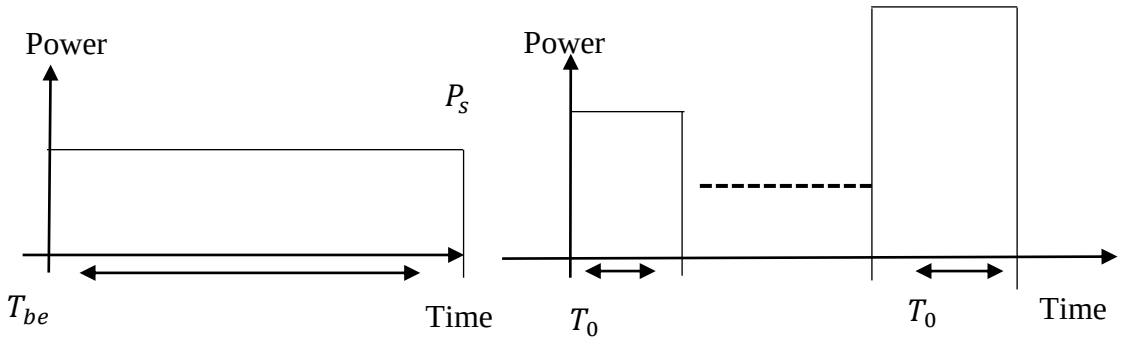


Figure 1. 11 *Task mapping in the system on chip working/ idle state*

1.5.1.1 Dynamic Voltage Scaling (DVS)

The authors in [87] CPU can function at a lower supply voltage when utilized at a lower frequency. A linear reduction mechanism is developed that can be measures the power consumption at the cost of a linear slowing in processor frequency as shown in Equation 1.17.

$$P_{spent} = ACV^2f \quad (1.17)$$

P stands for power spent *A* for activity factor, or the percentage of the circuit that is switching, ***C*** for capacitance, ***V*** for supply voltage, and ***f*** clock frequency.

The authors in [88] proposed a predictive dynamic thermal management for multiprocessor gradients that affect and imbalance the high peak temperature due to higher energy dissipation. The increase in cooling cost due to high temperature also affects the performance and reliability of the system to avoid these performance-related issues power management and thermal management. In [89] introduces the utilization bound $\propto$ that 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 $\geq \propto . m$ than $\propto$ utilization bound for P-EDF on the CPU is considered as:

$$u_{sum} \leq \propto\ m + 1/\propto +1 \quad (1.18)$$

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 have same utilization bound. Propose a novel dynamic power management scheme for adaptive pipelined MPSoCs, and utilization bound that is suitable for multimedia applications that finds the least number of processors that are required to schedule n task with u_{sum} [90] as shown below in Equation 1.19, 1.20 and 1.21:

$$m \geq \begin{cases} 1 \\ min\{[\frac{n}{\propto}] \end{cases} \quad if\ u_{sum} \leq 1 \tag{1.19}$$

$$m \geq \begin{cases} 1 \\ min\{[\frac{n}{\propto}] \end{cases} \quad if\ u_{sum} \geq 1 \tag{1.20}$$

$$m \geq \begin{cases} \frac{(\propto +1)u_{sum} - 1}{\propto} \} \end{cases} \quad if\ u_{sum} \leq 1 \tag{1.21}$$

1.6 Motivation

The high demand for performance and complexity in the design of MPSoC has increased the dissipation of energy. Recent studies have shown that the increase in energy and power consumption of multi-processor reaches linearly with the performance. Due to the high performance of embedded computing systems, the energy dissipation increases the overall on-chip temperatures, which affects the life span of the chip. These reliability and performance design concerns have put a special emphasis on low-power design. The emphasis and the demand for the energy-aware low-power design mainly in battery-powered portable systems due to portability and user convenience increases. To combat the high energy dissipation an energy-aware scheduling-based task migration policy to improve the reliability and user convenience using the low-power dynamic power management (DPM) technique for switching is introduced in this research.

1.7 Problem Definition

The most critical concerns in multi-core embedded systems are the performance and life span of the chip. The task scheduling and switching of jobs from one core to another are one of the major issues in today's MPSoC. High energy consumption due to higher task utilization and improper task ($\boldsymbol{\tau}$) scheduling is the most critical concern in MPSoC that reduces the overall performance, reliability and life span of the chip. High energy utilization increases the on-chip temperature which reduces the life span of the chip. It

also affects the reliability (hotspots, thermal cycles) as well as lowers the speed. Improper task ($\boldsymbol{\tau}$) scheduling increases the risk of tasks missing their deadlines, particularly in embedded systems that cause multiple performance and reliability issues. The key design difficulty in a task migration-based system is an accurate forecast of the processor's energy ($\boldsymbol{E_i}$), least used core, utilization factor ($\boldsymbol{u_i}$) and the workload $\boldsymbol{\tau}$ that needs to be relocated on an individual CPU.

1.8 Research Objectives

This thesis aims to reduce the consumption of energy from multiprocessors. To fulfill this aim, the following research objectives have been formalized:

The first objective of this research is to improve task scheduling by introducing an efficient task migration policy based on utilization factors using CPU core configurations for the mapping of tasks.

The second objective of this research is to improve the life span of the chip by optimizing the energy by proposing an optimal Energy-Aware EA-EDF scheduling based on DPM that balances the task load using intelligent core switching and task migration technique.

1.9 Main Contributions of the Thesis

The contributions of this thesis are well accomplished and summarized as follows:

1. Proposed a migration strategy for task migration technique for multiprocessor as discussed in (Chapter 3).
2. Core configurations for intelligent core switching and task migration are introduced. We have addressed the tasks that have no such restrictions to migrate to the core with less utilization.
3. Proposed an efficient energy-aware scheduling technique based on dynamic power management (DPM) for CPU switchnig that considers periodic task sets with implicit deadlines and arbitrary response times.
4. The proposed system has very efficiently reduced the consumption of energy using DPM based EA-EDF as discussed in (Chapter 4) and its complete evaluation experimental setup and improved results are discussed in (Chapter 5).

1.10 Used Environment

This complete model is simulated in a simulation tool for a real-time multiprocessor (STORM) using the hardware and software architecture of MARVEL INTEL PXA-270 MPSoC environment using ECLIPSE NEON on a research workstation with Intel UHD Graphics Xe 32EU-4300CU, 16GB RAM, 64-bit operating system, and 2.50 GHz Processor. The proposed model is effectively simulated.

1.11 Structure of Dissertation

In the 1st chapter, introduces the related work a detailed review of the previous work is carried out. Energy dissipation, background, objectives, and contributions to MPSoC are elaborated as well.

In the 2nd chapter, the related literature work is explained, and a detailed review of the previous work is carried out. The key feature and limitations of the previous work are summarized in tabular form. Various latest trends and different MPSoC-based power management system techniques are also deeply analyzed and key factors are elaborated as well. Methods used for the optimization of energy are reviewed critically for better understanding. The latest articles related to hardware and software-based energy optimization techniques mainly DPM, scheduling, DVFS, OS level, Compiler level and DTM are summarized.

The 3rd chapter of this thesis is based on the framework used for the migration of tasks to the cores. A workload model and a novel core switching model based on task migration that is further validated by the full task migration strategy for workload in the second part. The third part is about task utilization. The fourth part validates the task migration policy that integrated into the proposed EA-EDF based on the DPM strategy.

In the 4th chapter, an optimal DPM based energy-aware task scheduling mechanism is proposed based on the same full task migration policy of Chapter 3 is adopted. Similarly, this contribution is also further validated by the proposed system model in the next part. The core selection is also utilized using EA-EDF algorithms. An Energy/power model is formulated mathematically for energy optimization in the proposed energy-aware strategy including the core configurations and the algorithm is also introduced.

In the 5th chapter, the validation of the proposed model using the testing/experimental setup and simulation results and discussion. The last two sections are about the model validation of multiprocessors on the various proposed configuration in chapter 4.

In the 6th chapter, a discussion of the conclusion is drawn from this research, and its extension as future work is proposed.

CHAPTER 2
LITERATURE REVIEW

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

A variety of research in the field of energy and power-aware multiprocessor real-time scheduling and its related work is extensively discussed in detail. Various energy optimization methods used in MPSoCs are critically reviewed. We have also reviewed some of the previous fundamental research on task migration policy in a real-time multiprocessor environment and various research issues are described that are associated with the higher dissipation of energy.

2.2 Related Works

In [91] introduce an efficient energy dissipation strategy. Shrinkage of chip and certainly increasing the electronic element transistor on a chip the frequency and power densities are gradually increasing that causing many problems like power consumption and thermal issue as well as higher dissipation of energy. In [92] efficient multi-core systems are introduced but due to the increase in energy thermal issues arise that are the major problems. The authors in [93] introduce migration policies including the electro-migration process and dielectric breakdown process electro-migration process to reduce the effects of the increase in temperature on the chip and achieve a normal working condition [94]. Task scheduling and task allocation are facing issues during the migration of tasks to balance and manage power on multiprocessor systems [95]. In [96] introduced a dynamic technique for reducing the energy as well as thermal upsurge on each core. This energy aware mechanism for scheduling various tasks by considering their utilization for allocation of the task to a scheduler and various frequency and voltage levels based on the utilization factor of a task based on the EDF scheduling algorithm. The authors in [97] proposed an energy and thermal control technique that reduces errors by around 1.63% and considers hotspots by accurately measuring the temperature of the core. The below section describe the taxonomy for energy and power for an embedded system is shown below in Figure 2.1. We have divided a set of hierarchical categories of energy optimization techniques of MPSoCs into two main categories software-based techniques and hardware-based techniques.

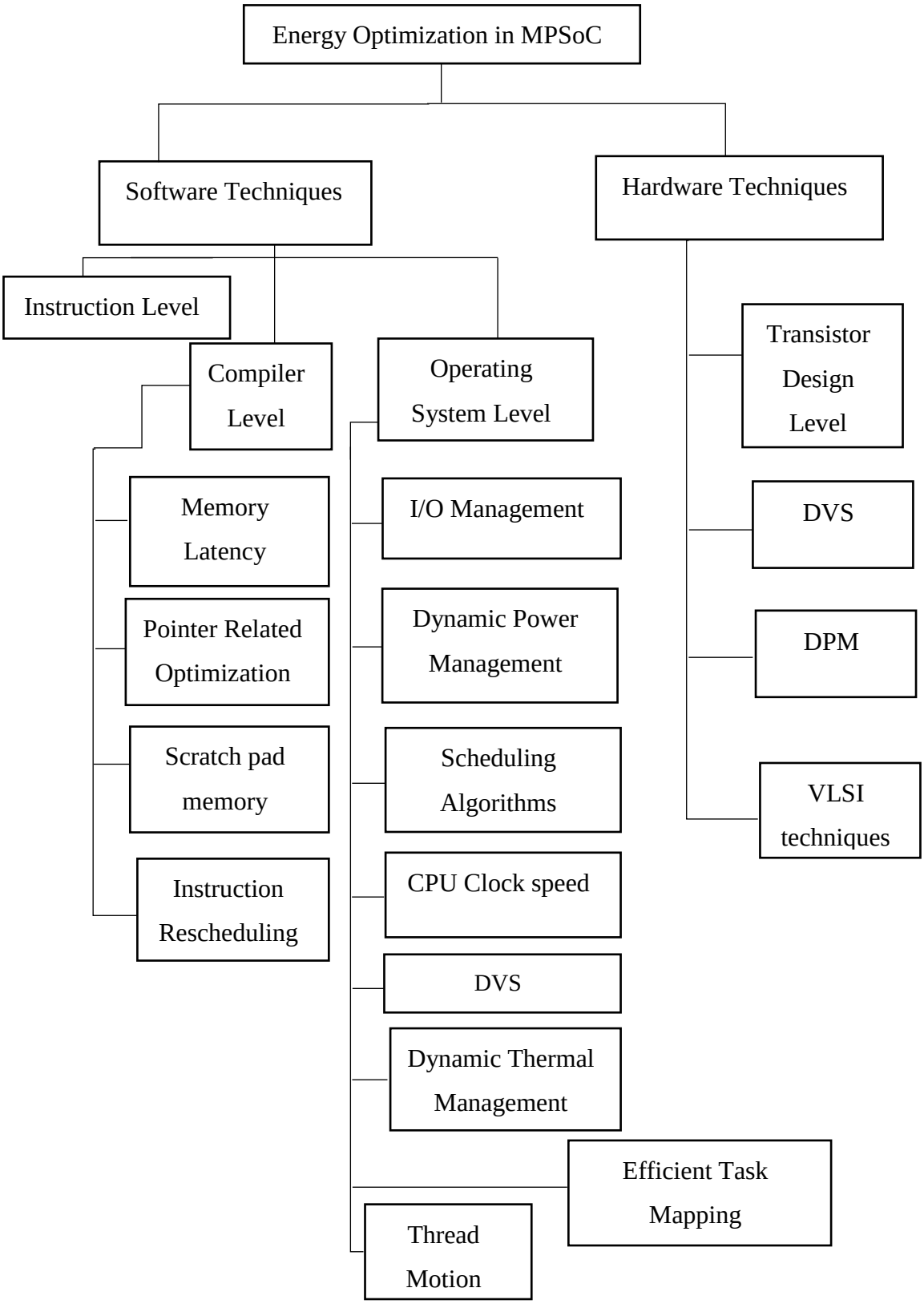


Figure 2. 1 *Energy/Power optimization techniques in embedded MPSoCs*

The hardware base energy reduction techniques are applied directly on the hardware chip that is not dependent on the system-level application as well as the operating system

(OS). Dynamic power management (DPM) and low power dynamic voltage scaling (DVS) are some of the wide used hardware-based energy optimizations techniques used in MPSoC [98]. Hardware-based techniques are implemented initially when the chip is in the developmental stages of manufacturing and designing. Moreover, the VLSI technique is widely used as a hardware-based technique for optimizing the energy and power of the different circuits using different levels specifically on the following levels i.e. logic, software, behavioral, system, and architectural level [99].

Hardware-based techniques are implemented on the system level in MPSoC. The 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 the software-based methods includes instruction level, operating system (OS) level and compiler level implementation.

2.2.1 Instruction Level

The authors in [100] introduce an instruction-level energy/power optimization technique. The instruction level analysis is used to assign a defined energy budget to any ready task.

2.2.2 Compiler level

The authors in [101] introduce a compiler-level mechanism that considers the behavior of an application during execution and determines the no of instructions being executed, order as well as type and has an impact on the reduction of energy consumption. Various sub techniques that are widely used to reduce the consumption of energy and power are as follows:

2.2.2.1 Memory Latency Optimization

The authors in [102] introduce a latency hiding 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.

2.2.2.2 Pointer related optimization

The implementation of pointer-related optimization is based on a software-level approach. The technique is also used in hardware as pointer synthesis [103].

2.2.2.3 Pad memory optimization

The authors in [104] introduce pad memory as high-speed on-chip memory and are widely used to reduce the consumption of energy during run time.

2.2.2.4 Instruction rescheduling

The authors in [105] proposed rescheduling of instruction is one of the highly used compiler-level energy and power reduction technique that is used to enhance the performance as well as also removes the pipeline stalls that causes a delay in processing.

2.2.3 Operating system (OS) level

Task-based energy and power management (TBEPM) or operating system (OS) based energy and power optimization are one of the widely used operating system level (OS) based optimization techniques that utilize the operating system according to the requirement and demand of tasks relates to OS processes [106]. The OS level optimization technique is implemented on dynamic power management (DPM), CPU clock, scheduling algorithms, Input-output (I/O) and speed management [107]. The authors in [108] proposed a technique that is used for the smooth transition of one process of OS to other processes. The authors in [109] introduce a thread motion technique that was presented to improve the well-known DVFS. Two levels of V-F domains are required to increase the performance to use a small program. An improved thread motion mechanism that allows applications to migrate the task to the cores with higher or lower voltage and frequency settings. Thread motion is used to swap two running applications between cores one at a higher V-F setting and one when the core is stalled for I/O activity

2.2.4 Dynamic Power Management

The authors in [110] introduce the system-level dynamic power management (DPM) technique that reduces the consumption of energy by switching system components when they are not using any resources or in an idle state. Due to advancements in computational embedded devices and increasing multi-task execution, the dissipation of energy becomes the captious design constraint of system-on-chip (SoCs) over the past decade as it limits the performance, reliability and battery life.

The authors in [111] introduce an efficient policy that keeps both reliability and performance while considering power degradation within allowed limits. Designs of these policies are considered to be an active and burning research topic in the field of MPSoC while designing embedded systems few challenges occur such as size, cost, power consumption and reliability. In [112] describes that DPM mainly deals with the development of policies that analyze the run-time behavior of the MPSoC system. The authors in [113] proposed a mechanism for the fundamental problem in the

implementation of DPM techniques is the non-uniform workload during the execution of the task. To solve this problem DPM uses a predictive algorithm that predicts the future workload by using different predictive models.

The below-mentioned section covers several system-level DPM approaches to save energy. Energy is directly affected by temperature when task load increases from the threshold value temperature also increase DPM allows MPSoCs to minimize power and energy consumption by optimizing the dynamic power. A decrease in the frequency saves a considerable amount of power but causes performance degradation in the multiprocessor [114]. The authors in [115] introduce a DPM-based energy-saving technique for reducing the dissipation of energy by keeping the idle mode of the CPU to a sleep state (low-power) whenever the CPU is not in use. Table 2.1 shows various modes of power consumption for embedded multiprocessor that is widely used and adaptive to DPM and DVS-based policies.

Table 2. 1 *A comparison of DPM-based Intel Embedded MPSoCs*

Work	CPU	Frequency	Active Power	System Bus	Idle Power	Current
[115]	PXA-270	624 MHz	925 Mw	208 MHz	260 mW	770 mA
[115]	PXA-270	520 MHz	747 Mw	208 MHz	222 mW	630 mA
[116]	PXA-250	208 MHz	279 mW	208 MHz	555 mW	150 mA
[117]	PXA-250	520 MHz	747 Mw	208 MHz	483 mW	150 mA
[117]	PXA-270	416 MHz	570 Mw	208 MHz	186 mW	500 mA
[117]	PXA-270	312 MHz	390 mW	208 MHz	154 mW	380 mA
[118]	PXA-270	312 MHz	375 Mw	104 MHz	109 mW	260 mA

DPM is a design technology that reconfigures the whole computing system dynamically in such a way that requested services can be provided with the minimum number of active CPUs with suitable performance levels. Dynamic power management (DPM) for energy optimization in MPSoCs can be distinguished by the level at which they are applied as hardware base (HW-DPM) and software base (SW-DPM) schemes.

Hardware base DPM techniques depends upon different HW components and selectively turns off the idle CPU or switch the state of some CPU, between active and idle to save energy and can be categorized as dynamic system component deactivation (DSCD) and dynamic system performance scaling (DSPS). DSCD is further classified as partial dynamic system component deactivation, complete dynamic system component deactivation (CDSCD). (HW-DPM) based predictive, stochastic and timeout policies are summarized while on other hand DSPS is classified as DVFS, and resource throttling.

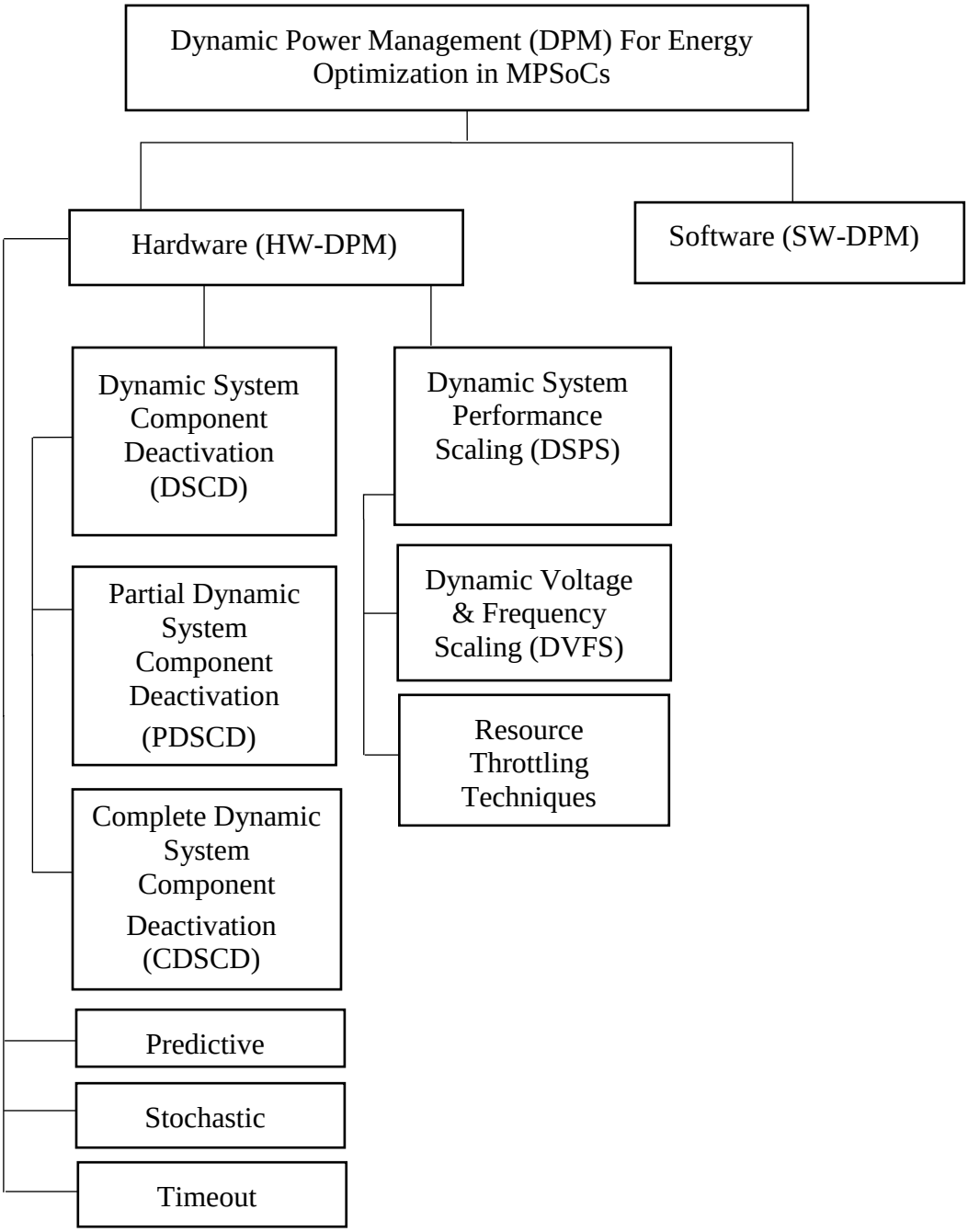


Figure 2. 2 *Taxonomy for DPM based high level energy management techniques*

In contrast, the SW-DPM techniques can be used to utilize the interface according to selected hardware based DPM policies.

2.2.4.1 Dynamic System Component Deactivation

The authors in [119] introduce an energy efficient DSCD dynamically moves the component to sleep mode from the running state into power-saving modes and brings them into idle and running state on demand to increase the efficiency of the CPU. The active, idle state, sleep states, and the power-off state are all possible power-saving states.

On average, all sleep stages utilize less energy than the active idle state. The amount of power dissipated by different sleep phases is proportional to the depth of sleep, i.e., the deeper the component sleeps, the less power is dissipated, but the longer it takes to wake it up. It's worth remembering that separate components can have distinct power states, and switching between them costs time and energy. The authors in [120] further classify DSCD partial dynamic system component deactivation in which clock gating of parts of an electronic component are considered while complete dynamic system component deactivation (CDSCD) can completely disable the components when CPU is inactive.

2.2.4.2 DPM based Predictive Technique

The authors in [121] introduce a technique that measures the length of the next idle period of the processor and efficiently reduces the consumption of energy. The decision can be quickly made when the processor is in sleep mode and this policy doesn't predict the length greater than or less than the break-even period (T_{be}).

Introduced an exponential average scheme that predicts the next idle period by considering an exponential average of the actual lengths of the previous idle period and the predicted one and reduces the higher utilization task that ultimately reduces the energy power dissipation for this an adaptive learning method is used to measure and encode the period of idle state. Predictive techniques get fail when the system scheduled load is changing rapidly or if there is no history for the past execution of a task. Prediction schemes can handle multiple power states [122]. The authors in [123] introduce another DPM based predication technique that works on the principle of calculating the length of the previous sequence of a user request that is known as a session. If the user is switching or shutting down the system the session length is reduced by using the adjustment factor the same factor is used to increase and handle the request from a user [124]. The authors in [125] proposed an energy aware prediction model that is working based on the sequence of shutdown commands for the system that is issued only when the system is idle for a long duration of time the session is predicted for the next task. This scheme predicts the future task/job load by creating a link between the current running workload and the previous load and predicts the load for future execution [126]. The authors in [127] proposed a predictive DPM technique that develops some relation between the past events of workload and future events by using the past and current information of workload events as timely predictions of task minimizes consumption of energy as well as power.

2.2.4.3 DPM based Stochastic Techniques

The authors in [128] proposed a DPM-based stochastic model that works for device power states that changes and measure the request arrival as stochastic processes. Minimizing delays and power dissipation is considered a stochastic optimization problem in modern MPSoCs. The proposed policy works on the basis of Markov process model that defines the system states. Markov model increases power saving by considering the performance. This policy considers various states of the MPSoC system that are deep sleep, sleep, standby, and running and idle.

The authors in [129] proposed a stochastic policy based on DPM for reducing the consumption of energy by measuring the idle intervals before the changing of state from higher to the lower energy state and task load considering the earliest deadlines and managed the performance using least no of cores. The authors in [130] proposed a Markov model based on stochastic control approach using DPM contains service provider, a power manager, service requester and cost metrics that supports in reducing the overall consumption of power and energy and enhances the performance.

2.2.4.4 DPM based Time Out Techniques

The authors in [131] proposed timeout technique based on DPM that change the state of the system into an idle state for running state when no task is executing on the system that period is equal to break-even time. The proposed policy is dependent on workload and these policies can improve safety by simply increasing the timeout values in this way the trade-off between safety and efficiency can be made by changing timeout values.

The authors in [132] proposed a DPM based predictive shutdown schemes that are based on a nonlinear regression using the history and threshold. The scheme is aggressive it shut down the system as soon as the system is idle the technique of predictive wakeup which considers the performance limitation which is paid on each wakeup. The main disadvantage of this scheme is that prediction must be accurate and energy and power wastage during the waiting period in case the prediction is wrong then the total consumption of power is greater than the energy optimized.

2.2.4.5 DPM based Dynamic System Performance Scaling (DSPS)

The authors in [133] proposed an energy aware DPM based hardware dynamic system performance scaling technique that is used in power-scalable components to dynamically alter CPU usage based on hardware. In DSPS the energy consumption is equal to the

product of CPU average power consumption and execution time. The authors in [134] proposed an efficient DSPS technique that dynamically increases the performance by changing its operating voltage and working frequency using DVFS and resources throttling mechanism that adjusts the CPU, GPU power, and energy consumption.

2.2.4.6 Software based Dynamic Power Management Technique (SW-DPM)

Hardware based DPM solutions are sophisticated. Its implementation and modification is very difficult. Therefore, the demand for software based solution based on DPM is widely used in modern MPSoCs. The authors in [135] proposed an advanced energy management (AEM) software based mechanism that uses input/output system (BIOS) level implementation for energy reduction.

The authors in [136] proposed an ACPI based offline DPM technique that is implemented on the operating system (OS) for configuration of CPU for low energy consumption mode in MPSoC. There are various offline and online algorithms that are associated with dynamic power management scheme. In offline scheduling algorithms, all assessments relating to scheduling are concluded before systems start execution. In this class of algorithms, time limits, period, tasks release time, WCET, AET and the order for execution of tasks are already known offline scheduling is generally used in a periodic style e.g. offline scheduling algorithms consist of table-driven scheduling and cyclic scheduling usually atmospheric applications are dependent on offline scheduling algorithms [137].

The authors in [138] proposed a dynamic technique for MPSoC that is used to control the gradual increase in on-chip energy and power consumption using offline algorithm by decreasing the running operations. This section summarized various offline algorithms that allow tasks to schedule and migrate by reducing the energy of the MPSoC by considering A-periodic and periodic task as shown in Table 2.2.

***Table 2. 2** A comparison of related work of Offline DPM Approaches*

Work	Algorithm	Speed Set	Periodicity	Dynamic Slack	Scheduler
[136]	HSCTB1	Continuous	A-per	No	Earliest Deadline First
[136]	HSCTB2	Discrete	A-per	Implicit	Earliest Deadline First
[137]	RHS	Discrete	Per	Implicit	Rate Monotonic
[137]	ES-RHS	Continuous	Per	Implicit	Rate Monotonic

The authors in [138] introduce an online scheduling mechanism in which the judgment process occurs at runtime irrespective of the work that yet needs execution. Scheduling decisions are very responsive to active jobs that are accessible at run time in distinction to the situation of offline scheduling algorithms. In [139] proposed online scheduling algorithm that considers those jobs whose performance is not known earlier. Their processing times are nameless and start working when the jobs are in a row. They turn into known only at the achievement of a job. Online algorithms are also designated as dynamic priority (DP) algorithms. The highlights of the related online DPM work are summarized in Table 2.3. The section summarized various online dynamic power management based algorithms that allow tasks to schedule and migrate while efficiently optimizing the consumption of energy.

Table 2. 3 *A comparison of related work of Online DPM Approaches*

Work	Algorithm	Speed Set	Periodicity	Dynamic Slack	Scheduler
[138]	HSCTB1	Continuous	A-per	No	EDF
[139]	LC-EDF	Continuous	Per	Implicit	Earliest Deadline First
[139]	LC-DP	Discrete	Per	Implicit	Rate Monotonic
[141]	ERTH	Discrete	Sporadic	Explicit	Earliest Deadline First

The authors in [142] introduce an efficient energy and power management technique known that dynamically adjusts the state of CPU that by decreasing the supply voltage to provide the required speed to the CPU to complete the task without missing any deadline. In [143] introduce a technique particularly useful for the memory-bound workload. This technique guarantee that all tasks will meet their deadlines requires some critical information about tasks such as release time, deadline, workload, and period and is a good way to save energy on the CPU. The computing burden on most microprocessor systems varies over time. It is preferable to run the CPU at the lowest frequency possible while maintaining the required performance level [144].

The authors in [145] proposed that due to the development of the chip size is decreasing and the number of transistors on a chip is gradually increasing and the dimensions of the chip are getting smaller than before. The increased temperature on the chip affects the reliability and performance of the Circuit that is used in the embedded device. In [146] introduces various hardware-based methods for power and thermal aware energy-efficient scheduling and its observed that dynamic thermal aware energy efficient scheduling is more efficient than fan thermal packaging because it is more reliable and lower in cost than fan thermal packaging. In [147] introduces an energy management

technique is used to move the chip to lower power consumption mode to reduce power densities. In [148] proposed a mechanism for multithreading applications that have different temperatures on chip because different applications perform their execution differently depending on their processing requirements. Some application needs more intense CPU processing because they create more heat on the chip as compared to those applications that don't need intense processing. The authors in [149] introduce a technique that is used to decrease the energy and power consumption of the chip by turning off all the cores that are in an idle state. This technique doesn't consider spatial gradients and reduces average power utilization and temperature on the chip by keeping the idle cores to be turned off when they are not in use. The below Table 2.4 highlights the related work that can be used as hybrid technique with DPM are as follow.

Table 2. 4 *A comparison of related work of hybrid techniques used with DPM*

Work	Algorithm	Speed Set	Periodicity	Scheduler
[150]	Online DVS	Continuous	A-per	Earliest Deadline First
[151]	ZMu	Discrete	Per	Earliest Deadline First
[152]	Online DVFS	Discrete	A-per	Rate Monotonic
[153]	Energy-saving LSP	Continuous	Sporadic	Earliest Deadline First
[154]	BSDVS	Discrete	Per	EDF/RM
[155]	BSDVF	Continuous	Per	Earliest Deadline First

2.3 Efficient Task Mapping & Scheduling

Due to advancement in embedded technology periodic activities shows more computational demand in a multi-core system these activities are categorized as periodic task set is continuously executing on definite rates with non-elastic deadlines, A-periodic task have very elastic deadlines and the sometime aperiodic task doesn't have any deadlines and sporadic task set usually have very hard deadlines and least arrival duration of a job in a task model that is already known by the processor [156]. The authors in [157] introduce a task migration mechanism that is used to move an executing task from one host CPU in a distributed architecture to another CPU. The selection of task and movement to the host CPU for a new task and the creation of the task on that host increase the performance, reliability and processing speed. Mapping and scheduling of tasks are one of the key design challenges in modern embedded systems this problem is to properly schedule tasks with the least consumption of energy while considering all the constraints. A schedule is considered proper if the deadline constraints are met. The authors in [158] introduce a mechanism that schedules a process generated by the

compiler to be loaded statically into the CPU or integrated as energy-aware into the real-time OS. Scheduling of tasks on a multiprocessor architecture is challenging as compared to scheduling on architectures with a single processor because a multiprocessor requires extra efforts for mapping tasks to the available processor. Moreover, deadline and precedence parameters the performance in case of improper task mapping [159]. The authors in [160] elaborate that the set of tasks belongs to the ready task set with implicit deadline parameters arriving on the CPU and each task required to be mapped to a processor but due to poor mapping of tasks it may lead to improper and infeasible schedule that affects the potential and increasing the energy consumption. Table 2.5 illustrates the comparative analysis of the proposed EA-EDF with current techniques as shown below.

Table 2. 5 *Comparative analysis of existing DPM based efficient scheduling*

Work	Approach	Contribution	Limitation
[161]	ERTH	Support discrete speed set and sporadic in nature used for soft real time systems	No task migration using A-periodic task
[162]	HSCTB1	The paper present an earliest deadline first scheduling based technique that used continuous speed set and efficiently schedule the task with earliest deadline	Supports task migration but the technique is only used in EDF scheduler and can be utilized for A-periodic task set only.
[163]	RT-DPM	The paper present a DPM based scheme that reduces the dissipation of energy and efficiently schedule and supports restricted task migration for sporadic task set of hard real time applications	The technique follows implicit deadlines but no task migration for periodic task set
[164]	EA-DVFS	Support energy aware dynamic adjustment of frequency at run time sporadic in nature mostly used for software base implementation of efficient task mapping	Limitation at real time migration of task using A-periodic and preemptive task arrived at same interval.
[139]	HSCTB1	This DPM based online scheduling technique supports task migration using A-periodic task	Don’t allow CPU allocates resources in preemptive environment also misses the deadline arrived at same time interval
[140]	LC-EDF	The paper presents an EDF based	Difficulty to implement for

		scheduling mechanism using dynamic power management techniques supports task migration using periodic task	aperiodic and sporadic task set in real time system because of energy overhead
[140]	LC-DP	Key growth in energy savings and adapts dynamic power management techniques effectively	No task migration using A-periodic task
[141]	LC-DP	The proposed strategy reduces consumption of energy and power and supports migration of task for independent periodic task set	No task migration using sporadic task
[165]	GEDF	This article focuses on maximizing the mapping of task by conserving the energy utilization in the multiprocessor and supports task migration for periodic task set	No task migration using periodic as well as preemptive task
[166]	TBP	Introduces a threshold based priority scheduling for MPSoCs that efficiently maps the A-periodic task set	No task migration using periodic and sporadic task
[167]	PDTM	The paper presents a predictive thermal aware task migration mechanism for reducing the on chip temperature by efficiently moving the task with higher utilization of periodic and sporadic task set.	Implementation in real time systems are difficult due to
[168]	Partitioned migration	This article focuses on efficient scheduling mechanism for mapping of ready task and statically assign the task to the CPU	No migration in case of uniprocessor architecture at system design time. Only low utilization processes specifically for one CPU.
[169]	Full migration	The paper presents an efficient scheduling mechanism that is less restrictive and widely used for multiprocessor scheduling. Execution of the task can halt on one CPU and reschedule the execution on a different processor.	Causes delay and when not utilized with EDF. Job-level parallelism is restricted because tasks cannot execute parallel on two or more different CPU cores.
[170]	Restricted migration	Proposed a restricted scheduling technique that migrates tasks between CPUs, each arrived task can execute on only one CPU core at a time.	In case the task required another core to complete the task the technique fails to migrate the executing task to the available CPU.

Scheduling algorithms are used to reduce the consumption of energy of the embedded system input-output (I/O). Low energy device schedulers and optimal device schedulers are widely used in MPSoC-based devices [166]. Therefore, an energy-aware efficient scheduling algorithm is needed to map each task to a processor in such a way that the entire task guarantees to meet the deadlines and also ensures a feasible schedule for the task. The priority characteristics of real time efficient task scheduling are discussed below:

2.3.1 Static Priority Scheduling

The authors in [171] introduce a priority scheduling mechanism in which all the tasks can change their priority at run time. They have static priorities set prior before the execution of job. In priority scheduling task can take decision of scheduling when the tasks are ready for execution. The structure for all temporal tasks is fixed and the scheduler can satisfy mutual exclusion and precedence e.g. round robin, offline and clock driven.

2.3.2 Dynamic Priority Scheduling

The authors in [172] introduces in this type of scheduling algorithm tasks have dynamic priority. Various priority stages can be allotted to the same task. Dynamic priority scheduling (DPS) is based on those task set that are in ready queue and mapping can be done at run time. In DPS scheduler requires computational resource parameters. Tasks with different jobs once move in running state cannot change its priorities. This means that once a job is set it is mandatory to complete its priority until the other scheduling event happens e.g. earliest deadline first and rate monotonic.

2.3.2.1 Earliest Deadline First (EDF) Algorithm

The jobs using EDF scheduling can have dynamic priorities. Scheduler can select those tasks that are in ready queue whose deadlines are coming next and very useful for multicore platforms. The utilization factor of the process is less than is approximately equal to 1 [173].

The authors in [174] introduces EDF in hard real time systems for scheduling of jobs that are preempt able and can attain 100% utilization for a central processing unit. Using EDF scheduling technique the utilization of central processing unit improves because of implication of dynamic priority. The priority of selecting each task depends on the fixed deadlines.

Table 2. 6 *Comparative Analysis of various related energy management scheme in MPSoCs*

Work	Technique	Key Contribution	Limitations
[175]	A-Symmetric cores	The asymmetric core aims to consider the big core for scalar program execution and a smaller core for parallel execution. Key growth in energy/power savings and adapts effectively to a wide range of software.	Less no cores are defined at design time that's why it's not cost-effective for the run-time task load. The data locality issue during the task migration affects the overall task scheduling.
[176]	Thread Motion	Proposed a thread migration scheme that allows migration of threads between cores is possible due to nearly two voltage/frequency domains quick switching between various voltage frequency settings.	Task migration is improper and the algorithm is not efficient in terms of energy and power saving.
[177]	Speculation Control	This speculation control scheme minimizes speculation to reduce energy consumption. Evaluating hybrid compatible techniques to get even greater performance and energy efficiency.	When used individually. it saves relatively little energy
[178]	DPM	The techniques of scheduling are controlling the performance and power levels of digital circuits and systems. Putting idle processors in a low-power state. DPM is a certain type of energy management technique. The performance of circuit's digital system is improved when power monitor to control the CPU modes during transitions to minimize the consumption of energy and power in MPSoC.	Don't allow processors to function at adaptable voltage and frequency levels. Threshold utilization of CPU levels for parallel loads is not properly mapped.
[178]	DVFS	This paper presents a DVFS-based technique to adjust the voltage and frequency to meet the performance and throughput. Reduction in voltage/frequency and energy level while reducing or limiting the performance impact. Energy and power saving is very effective with minimum performance loss and are simple to apply on a large scale.	The granularity cost of DVFS increases with long duration between power states

[179]	Variable Size	A comprehensive overview of the switching MPSoC resources as needed to conserve energy. It's easy to apply and can combine with other hybrid techniques.	Parallel loads affect the instructions due to improper core switching.
[179]	Core Fusion	The core fusion scheme is used to calculate the fewer cores that run parallel instructions and can be dynamically fused into a large core. Easy to execute the overall process.	Difficult to map cores in pipeline architecture.
[180]	SJF, RM Scheduling algorithms	Used to reduce the dissipation of energy of the embedded device. scheduling algorithms are used for optimization in low energy device.	Average waiting time and turn-around time is often quite long
[181]	Operating system (OS) level.	Reduction of energy and power using the operating system is defined as task-based management scheme. TBM utilizes the present information in the OS for managing energy and power.	Division of task load on the CPU causes the instruction execution delay.
[181]	DTM	Proposed a DTM technique for reducing the heat and thermal gradients in MPSoCs.	Difficult to implement in real-time systems.
[182]	symmetry-Aware	Allocate threads to cores to optimize performance. The benefit of executing on a fast vs slow CPU core is determined by the type of program being executed.	The algorithm is not efficient in terms of power savings when the parallel load is implemented.
Proposed EA-EDF		The proposed scheme is based on an EA-EDF scheduling mechanism based on DPM using task migration to reduce the energy consumption of the MPSoC. The state of the core switching mechanism is based on DPM that switches that sleep state to idle and idle to run when required. The proposed method also uses the full task migration policy when the utilization reaches its threshold to migrate the task to the core that is not in use. DPM moves the remaining cores that are not in use to the sleep state to ensure a significant reduction in energy and power consumption and guarantees	Delay in scheduling when a task with the same deadline and priority arrive together and utilization of the entire cores in the configuration reaches its threshold.

	the scheduling of tasks by meeting their deadlines.	

2.4 Summary

This chapter has presented a variety of research in the field of energy and power-aware multiprocessor real-time scheduling. Energy-aware scheduling is a rich area of research we have touched and critically reviewed many interesting fields of multiprocessor scheduling to analyze the computational abilities of multiprocessor systems on a chip. Moreover, various research issues are described that are associated with energy/ power dissipation and various characterization techniques for MPSoC-based embedded systems, initially various thermal approaches are discussed in detail. Various system-level characterization techniques for energy reduction mechanisms are discussed including, Dynamic power management (DPM), hardware base (HW-DPM) and software base (SW-DPM) including (DSCD) partial dynamic system component deactivation, complete dynamic system component deactivation and dynamic system performance scaling. Various hardware based-DPM techniques including predictive, stochastic, timeout policies, resource throttling, DVFS are summarized. Scheduling algorithms based on online and offline DPM have thoroughly discussed these techniques were chosen because these are the existing techniques and are related to the research presented in this thesis. In the end, a detailed conclusive comparative analysis of related work on energy/power management techniques in MPSoCs is reviewed in detail. The next chapters present an improved task migration policy for accurate task scheduling in multiprocessors under full task migration strategy and core configuration based on the proposed EA-EDF scheduling algorithm.

CHAPTER 3
AN IMPROVED TASK MIGRATION & CORE CONFIGURATION SELECTION TECHNIQUE

CHAPTER 3

AN IMPROVED TASK MIGRATION & CORE CONFIGURATION SELECTION TECHNIQUE

3.1 Introduction

We have explored an efficient full task migration policy that is integrated into the proposed EA-EDF scheduling technique using a core configurations model for a uniform multiprocessor platform based on utilization factor. Therefore in this chapter, we proposed an improved full task migration policy that migrates the task to avoid improper scheduling and consumes less energy by improving the scheduling of tasks that guarantees the entire task meets its deadline. Moreover, the complete framework for the task switching approach employed using full migration for MPSoCs is introduced.

3.2 Task Migration policy

Task migration is a mechanism that is used to move an executing task from one host CPU in a distributed architecture to another CPU. The selection of task and movement to the host CPU for a new task and the creation of the task on that host increase the performance, reliability and processing speed. The lack of task migration on time can affect the overall system performance [183]. Most of the current techniques improve energy and power management-related problems. These approaches have various research gaps including improper task migration as well as delay in migration and complexity because of high energy dissipation [30, 85, 153, 154] improper core switching and task scheduling [146,150], and lack of accurate prediction for task utilization and limitation of missing deadlines [149, 151] and no mapping for core configurations [145].

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

3.2.1 Partitioned Migration Scheduling

The authors in [184] introduce a partitioned migration scheduling technique, in which each ready task is mapped to a single processor *m* at system-design time. Statically assignment of the task to the CPU is preferable for the uniprocessor schedule as illustrated in Figure 3.1.

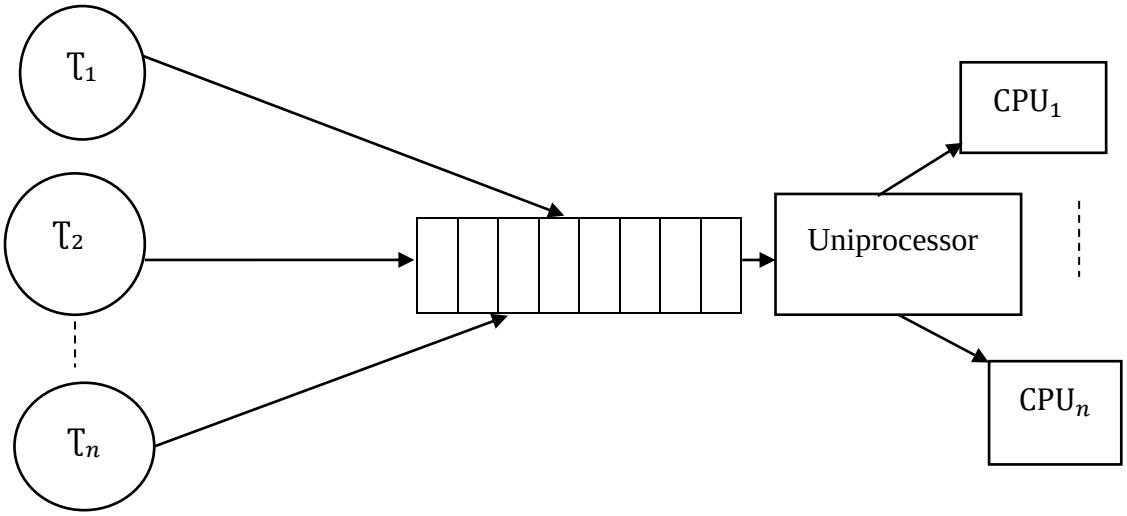


Figure 3. 1 *Scheduling of tasks using partitioned Migration* [184]

3.2.2 Full Migration Scheduling

The authors in [185] introduce full migration scheduling as the least restrictive technique widely used in multiprocessor scheduling. In this technique the execution of the task can 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.

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 Figure 3.2.

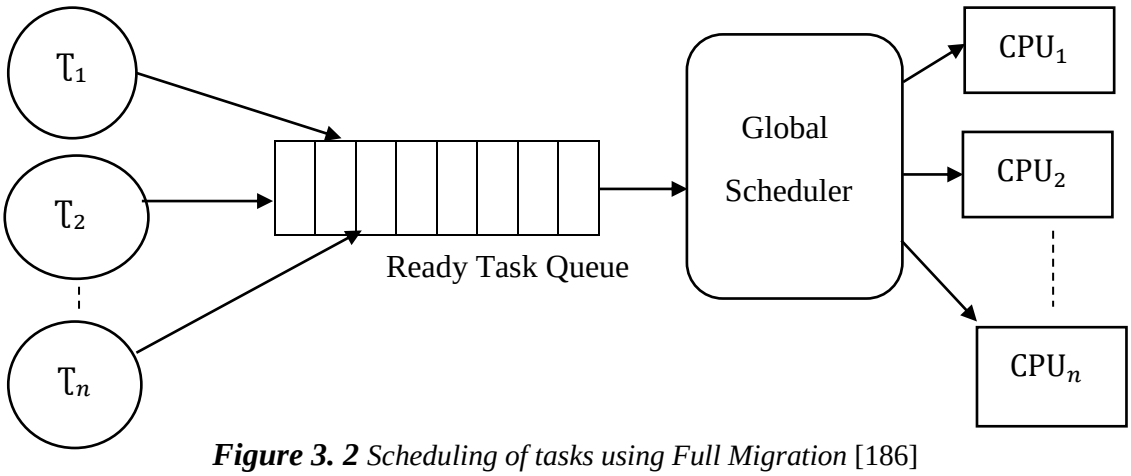


Figure 3. 2 *Scheduling of tasks using Full Migration* [186]

3.2.2 Restricted Migration Scheduling

The authors in [187] introduce a restricted migration scheduling that migrates tasks between CPUs, each arrived task can execute on only one CPU core at a time. The authors in [188] introduced an online migrations policy, but the arrivals of the task are not

prioritized. In restricted migration, a scheduler can adapt two levels. Generated tasks are placed on the global priority queue. A scheduling algorithm that is mainly used for uniprocessor to schedule that assigned task to CPU as shown in Figure 3.3.

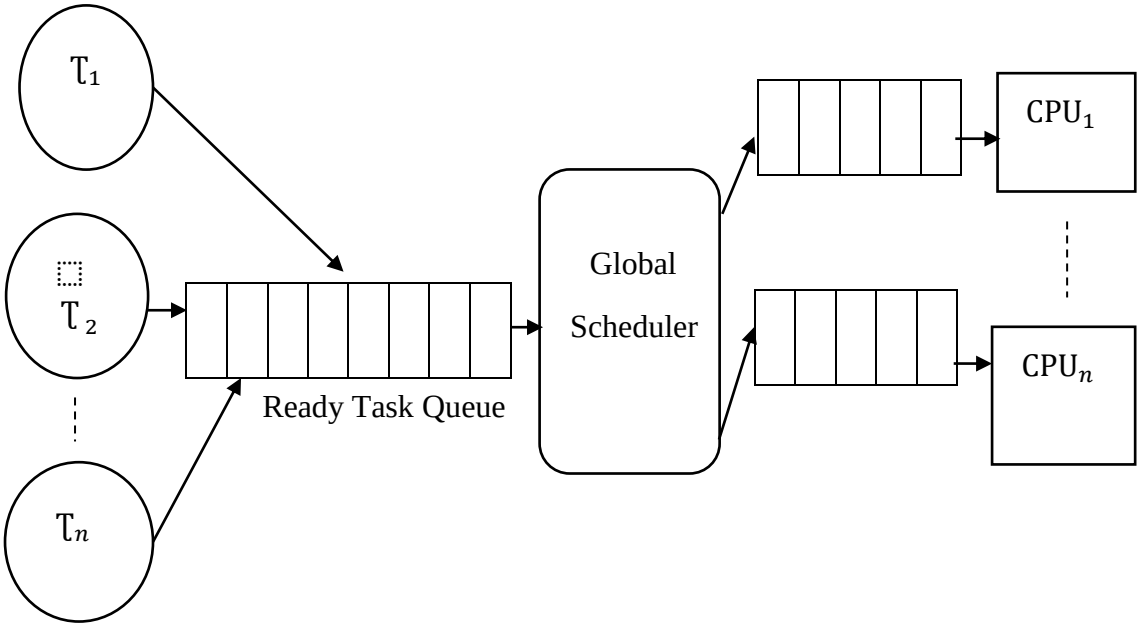


Figure 3. 3 *Scheduling of tasks using Restricted Migration* [189]

The first concern of existing models is that reducing energy consumption using any suitable task migration policy and switching approaches for the periodic task is not properly employed. The second concern is that the various energy and power management models that exist have their drawbacks under diverse conditions as demonstrated in Table 2.6. Our proposed EA-EDF scheduling technique using task migration improves the performance by reducing the consumption of energy and the utilization factor. We investigated that some researchers have worked on a DVFS partitioned, restricted task migration-based core selection approach for reducing the scheduling of tasks. In this research work, we presented a novel energy-aware low-power scheduling technique based on DPM for multiprocessors that considers periodic task sets with implicit deadlines and arbitrary response times in which various configurations for intelligent core switching and task migration are introduced. We have addressed the tasks that have no such restrictions to migrate tasks using full migration to the core with less utilization. The proposed system has very efficiently reduced the consumption of energy using EA-EDF.

In this regard, we proposed an energy-aware policy followed by task migration for optimization of energy and power using the DPM policy. This model not only migrates

task using the full migration technique but also provide the base for the EA-EDF scheduler to improve performance by reducing the consumption of energy and power.

3.3 Proposed Framework for Task Migration

In this section the proposed framework is used for the migration of tasks to avoid delay and guarantees all the ready task $\boldsymbol{t_i} \in \boldsymbol{\tau}$ schedule to a processor without missing their deadline for this framework for migration of task. Task migration is a technique used for proper scheduling of tasks that effectively works with DPM to reduce the consumption of energy while improving performance. Power and energy optimization on multi-core systems are developed to address MPSoC dissipation concerns. The influence of tasks may be recognized when looking for the best solution task factors can be examined as tasks have an impact on each other. An energy-aware EDF algorithm based on DPM suing task migration policy that optimizes energy while considering different configurations for migration of load is proposed in this research work. Normally, task schedules on each core are independent.

3.3.1 Proposed Workload Models

Throughout this dissertation, the real-time task t_i is denoted by a five-tuple. Each task $t_i \in \tau$ consists of $(a_{i,}, c_i, d_{i,} a_{i,} p_r)$ where the interval as release of the task r_i represents the time at which the $t_i \in \tau$ is in the ready queue, while c_i is the worst-case execution time, the deadline is denoted as d_i and $a_{i,}$ represents the time when the $t_i \in \tau$ scheduling request arrives while p_r represent the priority of the task. The task t_i arrive at the r_i release time when initially the system switched on and start execution for E_r for the time $a_{i,} a_{i,} + d_i$ referred as the time interval t_i Whereas $a_{i,}\ is\ R \geq 0$ {0, 1, 2, 3, …, n} while considering $d_i\ and\ E_r$ as $R \geq 0 = \{x \in \mathrm{R}: x \geq 0\}$. Task $t_i \in \tau$ is considered in the execution state if $t \in \tau[a_{i,} a_{i,} + d_i]$ and t_i is currently unfinished. We represent an *R-T* instance, $I = \{\mathrm{t}_1, \mathrm{t}_2, \ldots, \mathrm{t}_\mathrm{n}\}$, as a set of collection of periodic task τ assuming the task order according to their arrival time (i.e, for $t_1, t \in I: i \leq \tau\ if\ A_i \leq A_\tau$).

$\acute{\mathrm{O}}_f(I)$ Represents an *R-T* instance for each task $\mathrm{t'}_i$ in instance $I \in \acute{\mathrm{O}}_f(I)$. A task t_i in I instance considering same deadline and release time interval. The e_r execution of t'_icannot exceed the e_r execution of t_i. Therefore, $\mathrm{I} = \{t_1, t_2, \ldots, t_n, \ldots\} \in \acute{\mathrm{O}}_f(I)$ if $\forall t'_i \in (\mathrm{I}): (d'_{i,} d_i)\hat{}(a'_i, a_{i,})\hat{}\ (e'_{r,} e_r)$. An optimal scheduling algorithm proposed in chapter 4 guarantees all task $t_i \in \tau$ to meet their deadlines and enhances the performance

of MPSoC. Figure 3.4 illustrates the EA-EDF schedule the instance $I = \{t_1, t_2, \ldots, t_n, \ldots\}$, containing the periodic task $t_1 = \{0,1,3\}, t_2 = \{0,3,5\}$, $t_{1,1}$ represents the first task of t_1 with deadline $d_i = 3$ with higher priority than $t_{2,1}$while $t_{2,1}$ represents the first task of t_2 with deadline $d_i = 5$ having higher priority than $t_{1,2}$ also considering t_{12} has lower priority of $t_{2,1}$.

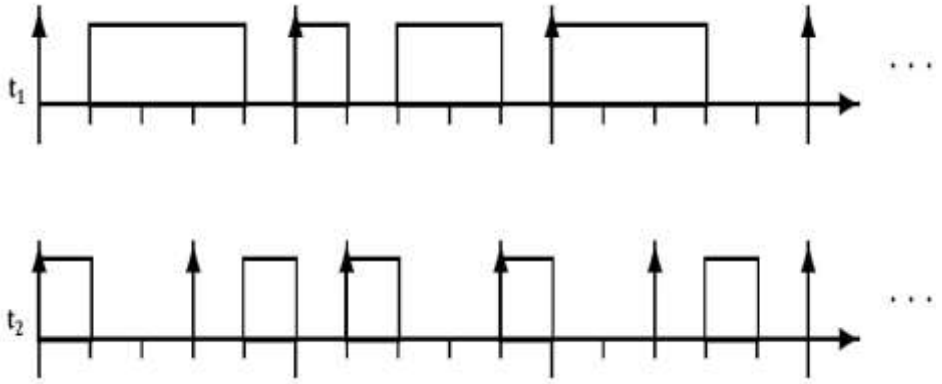


Figure 3. 4 *Scheduling of tasks using EA-EDF*

Table 3. 1 *Symbolic representation of various parameters in multiprocessors*

Symbols	Description
S_p	Schedule of processor
p_m	Denotes processor
m	Denotes the no of cores
$m(p_m)$	Total no of CPU in p
$s_{p(i)}$	Speed of i_{th} processor
$s_{(p_m)}$	The average processing power of the processor
$s_{i(p_m)}$	The average processing power of the i_{th} processor
$P_{running}\ (\alpha)$	Represents the processor executing τ denoted by α
$P_{idle}\ (\Upsilon)$	Represents the m processor in idle state denoted by Υ
$P_{sleep}(\beta)$	Represents the m processor in sleep state denoted by β
λ_m	Represents the core or processor m configuration by λ

3.3.2 Workload Utilization

In the proposed model an effective characterization of the R-T workload for periodic task set τ, the total u_i system utilization is measured using Equation 3.2.

$$System\ U_{total}\ (\tau) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \frac{c_a}{p_a} = u_i \tag{3.2}$$

Considering the utilization u_i all the tasks in τ will guarantee to meet all their deadlines. For migration of tasks the maximum utilization and total utilization are considered because periodic tasks are categorized based on u_i. The task t_i switches its core at maximum utilization that is measured using $U_{maximum}\ (\tau) \stackrel{\text{def}}{=} max_{t_i \in \tau}\{u_i\}$, while the total utilization is required to maintain the execution of the multiprocessor $U_{total}\ (\tau) = \sum_{i=1}^{n} \frac{c_a}{p_a} = u_i$. We combined all the tasks set into the same class and represented as $¥(U_{total}, U_{maximum})$ any t_i in the task set $\tau \in ¥(U_{total}, U_{maximum})$ will meet its deadline and all the task t_i complete its execution on or before the deadline if scheduled using the EA-EDF scheduling algorithm. If system-utilization $(\tau \leq m)$ then all the task τ are feasible and ready to schedule on an m -processor platform. EA-EDF gives a proper valid schedule of tasks on the {RT} instance, $I = \{t_1, t_2, \ldots, t_n, \ldots\}$ and schedules it properly without missing any deadline on the hardware architecture of a multiprocessor Intel PXA-270.

Definition: The most critical concerns in multi-core embedded systems are the performance and life span of the chip. The task scheduling and switching of jobs from one core to another are one of the major issues in today's MPSoC. Improper task (τ) scheduling increases the risk of tasks missing their deadlines, particularly in embedded systems that cause multiple performance and reliability issues. Considering two uniform multiprocessors denoted by P_1 and P_2 and a R-T system instance, $I = \{t_1, t_2, \ldots, t_n, \ldots\}$ as a set of periodic task τ, let's assume the total speed of the multi-processor $S_o = s_{p1(P)}$ and $s'_{p2(P)}$ are the schedules of instance $I = \{t_1, t_2, \ldots, t_n, \ldots\}$, on processor p_1 and p_2 with speed $s_{p(1)}$ and $s_{p(2)}$ on the multiprocessor P_1 and P_2 using proposed energy-aware scheduling guarantees the full migration of task τ according to the core configuration when the workload $t_i \in \tau$ exceed the threshold utilization factor u_i. All the requirements of $t_i \in \tau$ is required for measuring the energy and power of MPSoC for this purpose the utilization of task t_i is required for the migration of t_i.

The proposed system model is denoted as $Ø_i\{EAEDF, P, s_{p(i)}, I, t_i, \tau\}$, let $t_i$is a task belongs to the ready task set τ, I represent any {RT} instance, $I = \{t_1, t_2, \ldots, t_n, \ldots\}$ and $P = [s_1, s_2, \ldots, s_m]$ having m-processor and time $t \geq 0$, then $Ø_i$

$= \{\text{EAEDF}, \text{P}, \text{s}_{\text{p(i)}}, \text{I}, \text{t}_\text{i}, \tau\}$ indicates that the $\text{t}_\text{i} \in \tau$ is executing on a $\text{s}_{\text{p(i)}}$ of P at time $\text{t} \geq 0$ perfectly. The k utilization of task $\text{t}_\text{i} \in \tau$ i is denoted as $\text{u}_\text{i} = \frac{\text{c}_\text{a}}{\text{P}_\text{a}}$.

The above function Ǿ determines that all the $\text{t}_\text{i} \in \tau$ performed by scheduling algorithm EA-EDF on a given t_i or on RT instance $\text{I} = \{\text{t}_1, \text{t}_2, \ldots, \text{t}_\text{n}, \ldots\}$.

$$Ǿ(EAEDF, P, s_{p(i)}, I, t_i, \tau) = \begin{cases} 1, & \text{if EAEDF at } t_i \text{schedules to run on } \text{s}_{\text{p(i)}} \text{ at time t} \\ 0, & \text{otherwise} \end{cases} \quad (3.3)$$

For calculating the amount of work done (W) on all $\text{t}_\text{i} \in \tau$ of $\text{I} = \{\text{t}_1, \text{t}_2, \ldots, \text{t}_\text{n}, \ldots\}$ to determine the energy $\{\text{W}\{\text{EAEDF}, \text{P}, \text{I}, \text{t}_\text{i}, \tau\}\}, \{\text{W}\{\text{EAEDF}, \text{P}, \text{I}, \tau\}\}$,

Suppose t_i is a task belongs to the ready task set τ represents any {RT} instance, $\text{I} = \{\text{t}_1, \text{t}_2, \ldots, \text{t}_\text{n}, \ldots\}$ and $\text{P} = [\text{s}_1, \text{s}_2, \ldots, \text{s}_\text{m}]$ having m-processor using full migration technique that uses the earliest deadline first (EDF) scheduling algorithm for mapping and scheduling of periodic task set on a multicore CPU. The technique reduce and stabilize the energy as well as temperature of the multi-core system.$\{\text{W}\{\text{EAEDF}, \text{P}, \text{I}, \tau\}\}$, Below Equation 3.4 and 3.5 represents the total (joules) energy optimized using EA-EDF on task t of all {RT} instance I when time ≥ 0.

$$ǿ\{W\{EAEDF, P, I, t_i, \tau\}\} \stackrel{def}{=} \sum_{i=1}^{m} (s_i * \int_0^{\tau} Ǿ\{EAEDF, P, s_{p(i)}, I, t_i, k)dx\} \quad (3.4)$$

$$\{W\{EAEDF, P, I, \tau\}\} \stackrel{def}{=} \sum_{t_i \in I}^{m} W(EAEDF, P, I, t_i, \tau) \quad (3.5)$$

3.3.3 Proposed Full Task Migration Strategy for the workload Model

Scheduling algorithm applies migration of task $\text{t}_\text{i} \in \tau$ when multiprocessor system allows A $t_i \in \tau$ migrates if it starts running on one core of the CPU and later switches due to high utilization of task and migrates the task to the other core.

This dissertation considers the full migration strategy in which all the task $\text{t}_\text{i} \in \tau$ are allowed to migrate at any point as per the condition of $\text{u}_\text{i}$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. Figure 3.5 illustrate the full migration of task using EA-EDF. 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.

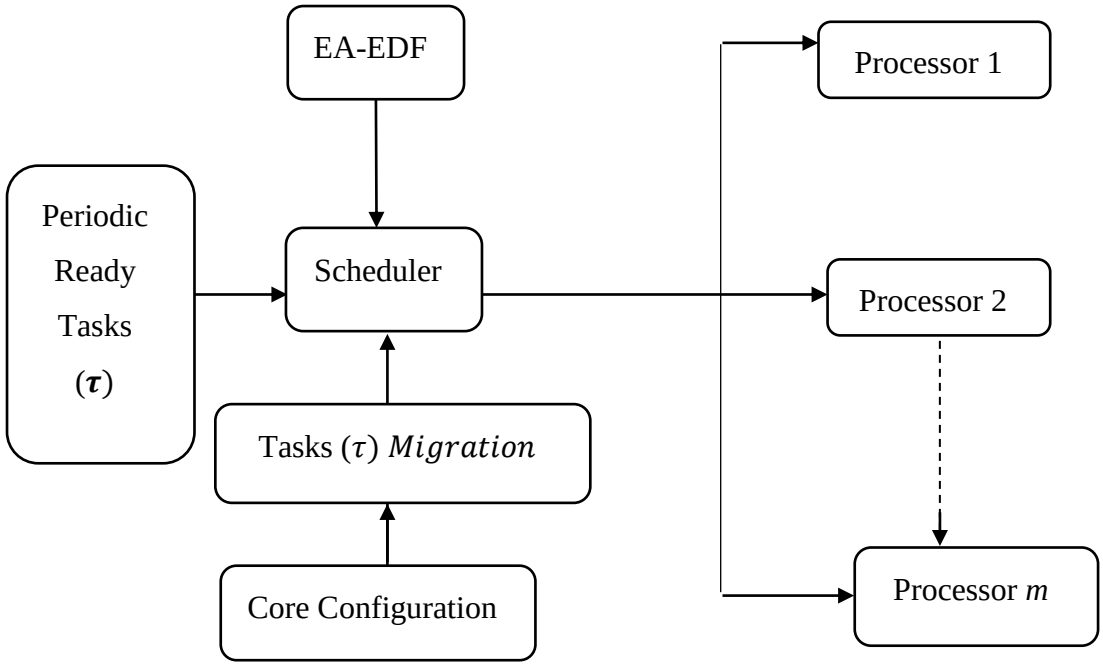


Figure 3. 5 *Scheduling with full migration using EA-EDF*

Considering two uniform multiprocessors denoted by P_1 and P_2 and a real-time (R-T) MPSoC system instance, $I = \{t_1, t_2, \ldots, t_n, \ldots\}$ as a set of periodic task τ, let's assume the total speed of the multi-processor $S_o = s_{p1(P)}$ and $s'_{p2(P)}$ are the schedules of instance $I = \{t_1, t_2, \ldots, t_n, \ldots\}$, on processor p_1 and p_2 with speed $s_{p(1)}$ and $s_{p(2)}$ on the multiprocessor P_1 and P_2 using proposed energy-aware scheduling guarantees the full migration of task τ according to the core configuration when the workload $t_i \in \tau$ exceed the threshold utilization factor u_i. All the requirements of $t_i \in \tau$ is required for measuring the energy and power of MPSoC for this purpose the utilization of task t_i is required for the migration of t_i . For all task $t_i \in \tau$ with the earliest deadline gets higher priority to be scheduled using this full migration strategy. It also determines where the task $t_i \in \tau$ execute in the processor. Moreover, it is observed that processor works faster when t_i has earliest deadlines.

Consider a task mainly periodic here $t_i \in \tau$ *where task* $t_1 = \{1,2,3\}$, $t_{1,1}$ represents the first task of t_1 , the speed of the multiprocessor is $s_1 = 2$ and $s_2 = 1$ $t_2 = \{1,3,4\}$, and processor is represented as $P = [s_{a(1)}, s_{a(2)}, \ldots, s_{a(m)}]$ having m-processor the value is P=[2,1] and $t_3 = \{0,6,8\}$. The peaks of the rectangle below the figure represent the processor speeds$s_{a(1)}, s_{a(2)}, \ldots, s_{a(m)}$. When a task $t_i \in \tau$ executes on $s_{1,}$, the corresponding peak of the rectangle is higher when it run on schedule of processor $s_{2,}$ the total area of a rectangle shows the requirement of execution.

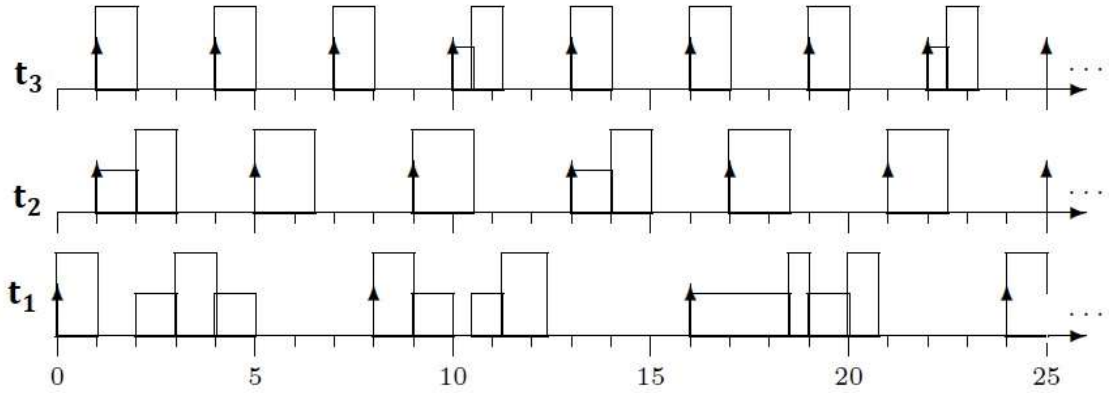


Figure 3. 6 *Task Scheduling for π= [2, 1] with full migration using EA-EDF*

Definition: 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 $s_{a(1)}$ and $s_{a(2)}$ represents 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 tasks 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 if it satisfies:

$$S(P) < (S_{a(1)}(P_2) \,.\, \acute{\emptyset}_i(P) + S(P_2) \tag{3.6}$$

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 INTEL PXA-270 (LEAT) 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 τload than the work function is represented as:

$$W(s_{a(1)}, P_1, I, t_a, t_a) < W(s_{a(2)}, P_2, I, t_{a=1}, t_a) \tag{3.7}$$

Equation 3.7 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 is shown in Equation 3.8.

$$W(s_{p(1)}, P_1, I, t_o) < W(s_{p(2)}, P_2, P, I, t_o) \tag{3.8}$$

Proof: If $t_o = r_{k,}$then the work done on P_1 for instant, I will increase as the arrival of the new task t_k increases the overall utilization of the selected configuration If (u_i) ≥max allowed τload, $r_{k,}$ is the peak task load period.

$$W(s_{p(1)}, P_1, I, r_{k,}) \geq W(s_{p(2)}, P_2, I, r_{k,}) \quad (3.9)$$

Therefore, the completion of $I = \{t_k, , \ldots, t_n\}$ of *RT* instance schedule in $s_{p(1)}$ at the 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 in running during that time interval and the remaining will be in an idle state

$$t_o - r_k = a_1 + a_2 + \ldots + a_{m(P)} \quad (3.10)$$

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:

$$W(S_o, P_2, I, r_{k,} t_o) \leq s_{p(1)}(P_2) * (t_o - r_k) \quad (3.11)$$

Task t_k can schedule and starts executing when the u_i reaches its threshold If $(u_i) \geq$ max allowed τ load enables the τ 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 job 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 $(\boldsymbol{\lambda_n})$:

$$W(S_o, P_2, I, t_k, t_o) \geq \sum_{k=1}^{m(P)-1} a_k s_k(P) \quad (3.12)$$

Schedule of a task Tasks set $t_k(t_1, t_2, \ldots t_n)$ $t_k$belonging to I is equal to $\sum_{k=1}^{m(P)} a_k s_j (P_m)$ during the interval $(t_o - r_k)$ the full migration (F-M) Schedule migratable $t_i \in \tau(a_k s_t)$ is shown in Equation 3.13.

$$\sum_{k=1}^{m(P)-1} a_k s_t(P_m) < (t_o - r_k) * S_o(P_2) \quad (3.13)$$

Combining Equations 3.4, 3.7, and 3.8 gives Equation 3.14 which shows at the instant I with a high task load instantly schedule on (P_m) CPU core at the speed of k_{th} the processor requires the switching of processor's $P_{sleep}(\beta)$ state to $P_{idle}(\Upsilon)$ in core configuration (λ_n) Switching (λ_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 high utilization task than $\delta(P_m)$ must be less than the load for proper scheduling of tasks.

$$\sum_{k=1}^{m(P_m)-1} \delta(P_m)) < s_{p(1)}(P_2) * (t_o - r_k) \quad (3.14)$$

Multiplying both sides by $\acute{\emptyset}_i(P)$ gives:

$$(\delta(P_m).\acute{\emptyset}_i(P_m)) = a_k.s_k(P_m) max_{1\leq 1\leq m(P)}\{\frac{S(P_m)-S_i(P_m)}{S_i(P)}\} \quad (3.15)$$

$S_k(P_m)$ is the average processing power of the k_{th} processor that is executing some task at maximum utilization of the core configuration is shown below in Equation 3.16.

$$\delta(P) max_{1\leq 1\leq m(P)}\{\frac{S(P_m)-S_i(P_m)}{S_i(P_m)} \geq \delta(P)\frac{S(P_m)-S_k(P_m)}{S_k(P_m)} \quad (3.16)$$

$$\delta(P)\frac{S(P_m)-S_k(P_m)}{S_k(P_m)} \quad (3.17)$$

Cancelation of the average power $S_k(P)$ when the full migration (F-M) schedules the migratable task ($(t_k \in I(a_k.s_k)$) at the schedule of the processor S_p whereas $S_k(P)$ is the average processing power of the k_{th} processor gives:

$$a_k.s_k(P_m)\frac{S(P_m)-S_k(P_m)}{S_k(P_m)} = a_k(S(P_m) - S_k(P_m)) \quad (3.18)$$

$$\sum_{k=1}^{m(P_m)-1} (a_k(S(P_m) - S_k(P_m) \quad (3.19)$$

$$< S_{p(1)}(P_2)\,.\,\acute{\emptyset}_i(P)(t_o - r_k) \quad (3.20)$$

Adding Equations 3.8 and 3.10 shows the avg processing p of the MPSoC decrease by adopting the task migration policy considering $(t_o - r_k) =$ 'E as shown in Equation 3.21.

$$\sum_{k=1}^{m(P_m)} (a_k(S_k(p)) + \sum_{k=1}^{m(P_m)-1} (a_i(S(P) - S_k(P_m))) < \quad (3.21)$$

('E)($S_{p(1)}(P_2)$. $\acute{\emptyset}_i(P) + S(P_2)$

Combining the sum $\sum$ of Equation 3.21

$$a_0(S(P_0)) + \sum_{n=k=1}^{m(P)-1} [a_1(+S(P_0) - S_0(P))] \quad (3.22)$$

$$< (\text{'E})(\ S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P) + S(P_2) \quad (3.23)$$

Select the $P_{exe}(\alpha)$ core resources in $(\lambda_1, \lambda_2, .. \lambda_n)$ for the migratable task (($t_k \in I(a_k.s_k)$) at the schedule of the processor S_p to least used suitable $\boldsymbol{\lambda}$ to achieve energy efficient $\boldsymbol{\tau}$ execution cancelation of $S_k(P)$ parameter gives:

$$a_m S(P) \sum_{k=1}^{m(P)-1} (a_k(S(P_m)) < (\text{'E})(\ S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P) + S(P_2) \quad (3.24)$$

Let's combine both the LHS the equation becomes:

$$S(P) \sum_{k=1}^{m(P)} (a_k(S(P_m)) < (\text{'E})\ (\ S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P) + S(P_2) \quad (3.25)$$

let's substitute equation 3.6 $t_o - r_k = a_1 + a_2 + \ldots + a_{m(P)}$ in the above equation gives

$$S(P) \sum_{k=1}^{m(P)} (a_k(S(P_m)) < (a_1 + a_2)\ (\ S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P) + S(P_2) \quad (3.26)$$

$$S(P) * (\text{'E}) < \text{'E})(\ S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P) + S(P_2) \quad (3.27)$$

By dividing the equation by $(t_o - r_k)$ the equation becomes:

$$\frac{S(P_m).(t_o - r_k) < (t_o - r_k)(\ S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P) + S(P_2)}{(t_o - r_k)} \quad (3.28)$$

Cancelation of ('E) proves that the workload efficiently migrates and guarantees the meeting of the deadline using the proposed migration framework ultimately reducing the consumption of energy by switching the cores to low power mode when the core is not running task (($t_k \in I(a_k.s_k)$)

$$S(P_m) < (S_{p(1)}(P_2)\ .\ \acute{\emptyset}_i(P_m) + S(P_2) \quad (3.29)$$

Based on the above equation its confirmed that using unrestricted full migration permits all the tasks to migrate among processors in the configuration if required. Moreover, our energy-aware proposed EA-EDF scheduling algorithm discussed in chapter 4 supports a full migration technique.

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 schedule time intervals$t > 0$: Section 3.2.4 elaborates on various stages of core configurations using the proposed task scheduling paradigm.

Table 3. 2 *Symbolic representation of various parameters used in task migration scheme*

Symbols	Description
P	Denotes processor
m_p	Total no of CPU in P
$d_{i(e)}$, $d_{i(l)}$	Denotes the earliest deadline, late deadline
$T_{1(F)}$,$T_{2(F)}$, $T_{n(F)}$	Fixed priority task
$T_{1(M)}$,$T_{2(M)}$	Migratable task

Definition: Considering uniform multiprocessor denoted by P_1, P_2 and P_m and a (R-T) system instance, $I = \{t_1, t_2, \ldots, t_n, \ldots\}$ as a set of periodic task τ, the schedules of instance $I = \{t_1, t_2, \ldots, t_n, \ldots\}$, on processor p_1 and p_2 having fixed task t_1, t_2 represented as $t_{1(F)}$ and $t_{2(F)}$ on the multiprocessor $P=\{P_1, P_2 \ldots P_m\}$. All the requirements of $t_i \in \tau$ is required for measuring the energy and power of MPSoC for this purpose the u_i of t_i task is required for the migration of task t_i using proposed energy-aware scheduling with task full migration of task τ guarantees the scheduling of $t_i \in \tau$ on the core configuration when the workload $t_i \in \tau$ exceed the threshold utilization factor u_i. If $(u_i) \geq$ max allowed threshold τ load value than the selection of task migration not only reduces consumption of energy also enhances the CPU performance by switching the states of CPU from P_{idle} to $P_{sleep}(\beta)$.

The schematic illustration of EA-EDF with task migration during the execution phase ensures that migrating tasks $t_{1\,(F)}, t_{2\,(F)} \in \tau$ never miss their deadlines are shown in figure

3.7. Task $t_i \in \tau$ with high priority H_p considered on priority for migration as compared to tasks with lower priority L_p.

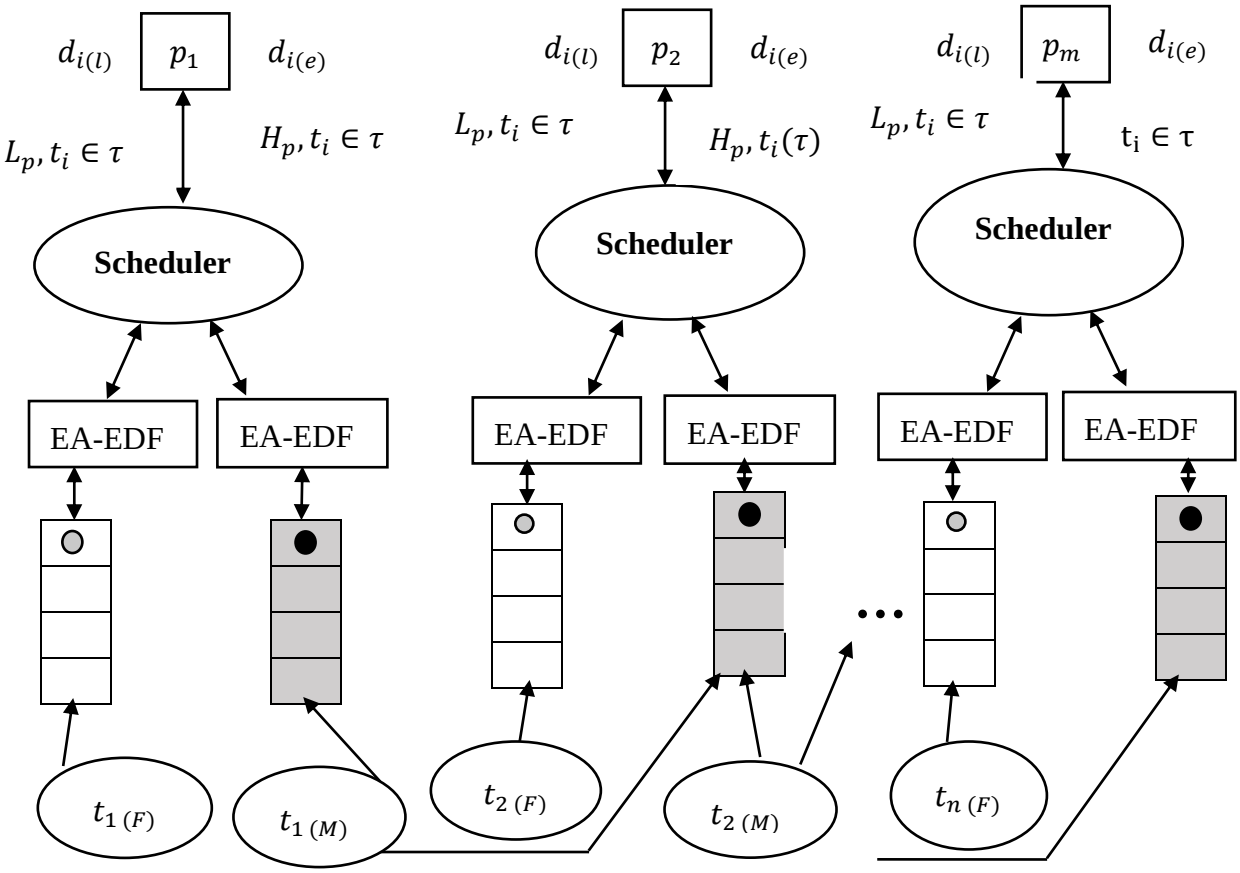


Figure 3. 7 *Representation of proposed full task Migration scheme during execution*

Full task migration is integrated into the proposed EA-EDF scheduling algorithm that is responsible to schedule tasks on configurations depending on the on-task workload utilization factor (u_i) on the cores. Selection of core uses the average number of tasks that are in running mode when tasks are in running condition the u_i remains constant but due to an increase in tasks the u_i of the multi-core processor is increasing.

All the tasks are permitted to migrate among available and least used processors. The below section elaborates the flow chart of the proposed task migration strategy that is integrated into the proposed scheduling mechanism.

3.3.4 Proposed Task Migration Frame Work & Stages of Core Configurations (λ_n) based on (u_i).

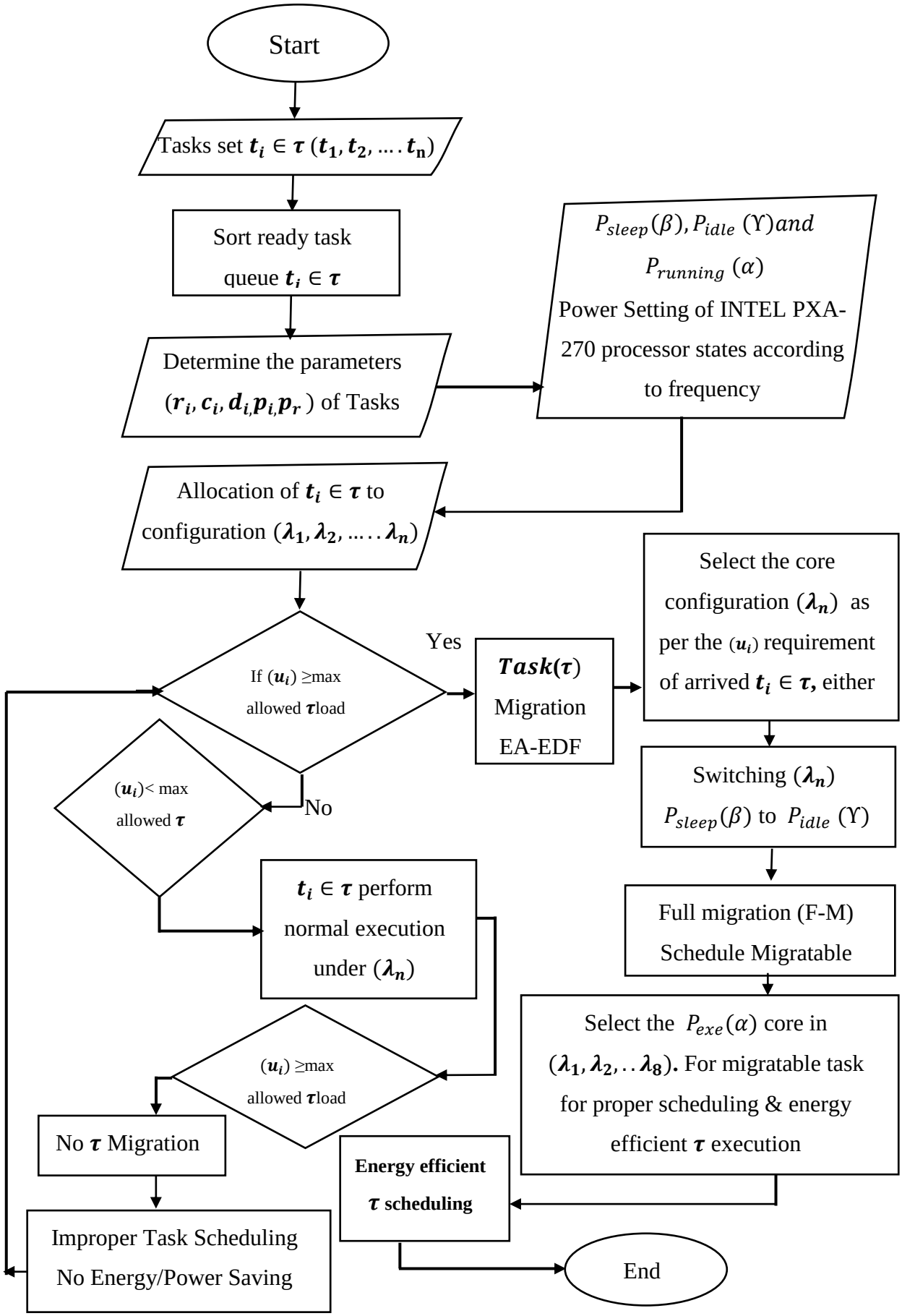


Figure 3. 8 *Flowchart of the proposed framework*

If $(u_i) \geq$ max allowed threshold τ load value than the selection of task migration not only decreasing the consumption of energy but also enhances the performance by avoiding the task delay during execution in the core configuration (λ_n) as per the (u_i) requirement of arrived $t_i \in \tau, P_{sleep}(\beta), P_{idle}(\Upsilon) and \ \ P_{running}(\alpha)$ power setting of INTEL PXA-270 processor states labeled as LEAT processor according to the standard frequency 624MHZMove the selected (λ_n) from $P_{sleep}(\beta)$ to $P_{idle}(\Upsilon)$ state to avoid delays and threshold workload value which is already set to reduce delays and migration cost and if a core reaches a threshold value will shift the workload to that core which is in less use are in sleep mode using full migration (F-M) strategy that migrates and schedule the ready queue tasks $t_i \in \tau$ into the selected core configuration (λ_n) without missing the (d_i) and save energy as well as reduce power consumption. In the next step, a CPU is allocated to perform energy-efficient execution as per the EA-EDF Scheduling policy to reduce the consumption of energy and power.

In case If $(u_i) \geq$ max allowed threshold (T) without opting for any τ migration policy doesn't schedule efficiently. Moreover not optimizing any energy and power that causes a delay in execution and misses the implicit (d_i). $\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_i}{p_i}$ $wherase\ c_i$ represents the WCET worst-case execution and p_i denotes the period, the researcher considers the hardware architecture of the octa-core INTEL PXA-270 (LEAT) multiprocessor. The octa-core processor can execute multiple tasks at a time each task on an individual core if the processor is executing some task is represented as $P_{exe} = \alpha\ while\ P_{sleep} = \beta$ denotes the sleep state of processor m_n. In the proposed system the migration of tasks depends on the utilization factor of MPSoC.

This section describes eight core configurations (λ_n) of the INTEL PXA-270 (LEAT) multiprocessor to run different tasks and the least used core λ_n have the highest priority to be selected first. Task migration ensures that tasks must meet their deadlines and the EA- EDF design ensures smooth execution of the task. Following are the various stages proposed for ensuring the task migration in the proposed (EA-EDF) scheduling algorithm as introduced in chapter 4. Table 3.3 show the various stages for core configurations required for the selection of CPUs.

Table 3. 3 *Various proposed stages for core configurations*

Scheduling Model	Stages	u_τ Formulations	Scheduling Paradigm	State of Core (α, β)
EA-EDF-based	$\boldsymbol{\lambda_1}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} = 1 - 9$ $U_\tau \leq m(p)+1$ $\lambda_1: (m_1)$ or (m_2) (m_3), or (m_4), (m_5), or (m_6), (m_7), or (m_8).	Full Migration	$\alpha = 1$ $\beta = 7$
EA-EDF-based	$\boldsymbol{\lambda_2}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} =$ $9.1 - 18\ U_\tau \geq m(p) + 2$ $\lambda_2: (m_1, m_2)$, (m_4, m_7), (m_3, m_5), (m_6, m_8).	Full Migration	$\alpha = 2$ $\beta = 6$
EA-EDF-based	$\boldsymbol{\lambda_3}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} =$ $18.1 - 27$ $U_\tau \geq m(p) + 3$ $\lambda_3: (m_1, m_2, m_3)$, (m_4, m_5, m_6), (m_4, m_7, m_8).	Full Migration	$\alpha = 3$ $\beta = 5$
EA-EDF-based	$\boldsymbol{\lambda_4}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} = 27.1 - 36$ $U_\tau \geq m(p) + 4$ $\lambda_4: (m_1, m_2, m_3, m_4)$, $(m_5, .., m_8)$	Full Migration	$\alpha = 4$ $\beta = 4$
EA-EDF-based	$\boldsymbol{\lambda_5}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} = 36.1 - 45$ $U_\tau \geq m(p) + 5$ $\lambda_5: (m_1, m_3, m_5, m_7, m_8)$, $(m_2, m_4, m_6, m_8, m_1)$	Full Migration	$\alpha = 5$ $\beta = 3$

EA-EDF-based	$\boldsymbol{\lambda_6}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} = 45.1 - 54$ $U_\tau \geq m(p) + 6$ λ_6: $(m_1, m_2, m_3, m_4, m_5, m_6)$, λ_6: $(m_1 m_2, m_3, m_6, m_7, m_8)$ λ_6: $(m_1 m_2, m_3, m_4, m_7, m_8)$	Full Migration	$\alpha = 6$ $\beta = 2$
EA-EDF-based	$\boldsymbol{\lambda_7}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} =$ $54.1 - 62.5$ $U_\tau \geq m(p) + 7$ λ_7: $(m_1, \ldots, m_7)$	Full Migration	$\alpha = 7$ $\beta = 1$
EA-EDF-based	$\boldsymbol{\lambda_8}$	$\forall\tau \rightarrow u_\tau = \sum_{K=1}^{n} \frac{c_a}{p_a} >$ 62.5% $U_\tau \geq m(p) + 8$ λ_8: $(m_1, m_2, \ldots, m_8)$	Full Migration	$\alpha = 8$ $\beta = 0$

3.4 Summary

This chapter has explored the full task migration policy based on the EA-EDF scheduling test for a uniform multiprocessor platform associated with the utilization factor to improve the scheduling of task and guarantees that the entire task meets its deadline. Framework for task switching approach that is employed using full migration for multiprocessor system on a chip.

A method for workload utilization is also developed and various task mappings are introduced to check the missing deadlines are evaluated using a proposed scheduling algorithm. Therefore this chapter proposed a full task migration policy that can be used to schedule real-time instants and consumes less energy and forces the system to migrate the task and ensure that all the task deadlines will be met.

The shortcomings of previous studies are addressed by opting a proper task scheduling and migration scheme. The significance and implementation of the various stages of core configurations are illustrated in chapter 4 section 4.3.5.

CHAPTER 4

AN OPTIMAL DPM BASED ENERGY-AWARE-EARLIEST DEADLINE FIRST (EA-EDF) SCHEDULING

CHAPTER 4

AN OPTIMAL DPM BASED ENERGY-AWARE-EARLIEST DEADLINE FIRST (EA-EDF) SCHEDULING

4.1 Introduction

This chapter contains a detailed overview of the proposed research methodology and components used in the evaluation that is adapted to propose an energy-efficient DPM-based scheduling algorithm (EA-EDF) to increase the performance of the system by reducing the energy dissipation using suitable abilities of the DPM policy and proposed full task migration policy as introduced in chapter 3.

4.2 Problem Statement

The most critical concerns in multi-core embedded systems are the performance and life span of the chip. The task scheduling and switching of jobs from one core to another are one of the major issues in today's MPSoC. High Energy consumption due to higher task utilization and improper task ($\boldsymbol{\tau}$) scheduling is the most critical concern in MPSoC that reduces the overall performance, reliability and life span of the chip. High energy utilization increases the on-chip temperature which reduces the life span of the chip. It also affects the reliability (hotspots, thermal cycles) as well as lowers the speed. Improper task ($\boldsymbol{\tau}$) scheduling increases the risk of tasks missing their deadlines, particularly in embedded systems that cause multiple performance and reliability issues. The key design difficulty in a task migration-based system is an accurate forecast of the processor's energy (*E*), least used core, utilization factor ($\boldsymbol{u_i}$) and the workload $\boldsymbol{\tau}$ that needs to be relocated on an individual CPU.

The objective is to decrease the energy consumption of the CPU using energy-aware scheduling for $\boldsymbol{t_i} \in \boldsymbol{\tau}$ over **m**-processor $\boldsymbol{P} = \{\boldsymbol{p_1}, \boldsymbol{p_2}, \ldots, \boldsymbol{p_m}\}$ to complete before the deadline by adopting the suitable task migration for switching to reduce the consumption of energy because in multiprocessor systems task switching across various cores is a prevalent problem because the destination target core may be in sleep mode there may be a delay while moving tasks from one core to another. Our major goal is to design a strategy to minimize the overall CPU's energy $\boldsymbol{E_{total}(p)}$ while the constraints timing interval especially the deadline of all the task $\boldsymbol{t_i} \in \boldsymbol{\tau}$ of applications are guaranteed using

the efficient scheduling mechanism based on task migration. Task migration is a technique used for proper scheduling of tasks that effectively works with DPM to reduce the consumption of energy while improving performance. Energy optimization on multi-core systems is developed to address MPSoC dissipation concerns. The influence of tasks may be recognized when looking for the best solution task factors can be examined as tasks have an impact on each other. An energy-aware EDF algorithm based on DPM using a task migration policy that optimizes energy while considering different configurations for migration of load is proposed in this research work. Normally, task schedules on each core are independent.

4.3 Proposed Framework

This section describes the complete framework of proposed EA-EDF techniques and framework parameters. We have proposed EA-EDF an optimal algorithm for the high energy dissipation of multiprocessor scheduling problems. Since the power consumption Pi(s)/s is increasing at each cycle for every task $\boldsymbol{t_i} \in \boldsymbol{\tau}$ in a given periodic task set, the CPU executes each task $\boldsymbol{t_i}$ of the task set $\boldsymbol{\tau}$ at some speed $\boldsymbol{s}$. For this, we have proposed EA-EDF an energy-efficient earliest deadline first scheduling algorithm based on the DPM technique using a task migration policy that guarantees to reduce the consumption of energy by meeting all the deadlines by monitoring the utilization factor ($\boldsymbol{u_i}$). An energy-efficient policy for various configurations of cores to predict the power and energy profiles using both hardware & software architecture of the Intel PXA-270 (LEAT) MPSoC. Scheduling technique based on task migration reduces energy and produces energy-efficient results according to the utilization factor ($\boldsymbol{u_i}$). The proposed model uses an EA.EDF scheduler for energy and power optimization and is independently demonstrated to be the best-performing technique. Considering a multiprocessor architecture with $\boldsymbol{m}$ cores with $\boldsymbol{P} = \{\boldsymbol{p_1}, \boldsymbol{p_2}, \dots, \boldsymbol{p_m}\}$ having a periodic task set $\boldsymbol{\tau} = (\boldsymbol{t_1}, \boldsymbol{t_2}, \dots, \boldsymbol{t_n})$ as a set of periodic task $\boldsymbol{\tau}$, over $\boldsymbol{m_{(P)}}$ processors where many tasks $\boldsymbol{\tau}$ have a common deadline $\boldsymbol{d_i}$and are ready at time 0. Each task $\boldsymbol{t_i} \in \boldsymbol{\tau}$ is linked with CPU requirement that is approximately equal to the energy and power consumption function $\boldsymbol{P_i(s)}$ and CPU-cycles $\boldsymbol{c_s}$ at the given CPU speed. Each task $\boldsymbol{t_i} \in \boldsymbol{\tau}$ consists of $(\boldsymbol{a_{i,}}, \boldsymbol{c_i}, \boldsymbol{d_{i,}} \boldsymbol{p_{i,}} \boldsymbol{p_r})$, where the release time or activation period of the task $\boldsymbol{a_i}$ it's the time when the $\boldsymbol{t_i} \in \boldsymbol{\tau}$ scheduling request arrives, represents the time at which the $\boldsymbol{t_i} \in \boldsymbol{\tau}$ is in the ready queue, while $\boldsymbol{c_i}$ is the worst-case execution time, $\boldsymbol{p_i}$ represents the period and

the deadline is denoted as $\boldsymbol{d_i}$. $\boldsymbol{p_r}$ represents the priority. The proposed policy operates with a set of parameters and generates effective outcomes for the given workload while also optimizing the chip's lifespan and improving QoS by lowering energy consumption. The task set generator provides randomized task sets under a ready task queue that are being used to generate workloads under various constraints as well as the number of CPU cores specified by the user in extensible markup language (XML) based on the application given to STORM as an input. EA-EDF the scheduler can also have a set of prepared tasks that are scheduled according to the scheduling policy by considering suitable migration of task policy that produces energy and power profiles on the individual cores according to the given set of parameters in XML.

A periodic task set $\tau = (\tau_1, \tau_2, \tau_3, \tau_4, \tau_5 \ldots., \tau_n)$ and a utilization factor that is non-increasing is represented as:

$u_{i=}u_{i+1}$ for all $i = 1, \leq i \leq n,$ (4.1)

where $u_{i=}\frac{\alpha_i}{p_i}$ consider $u_{i=}\sum_{j=0}^{i} = 1$ for $\forall i = 1, \leq$ i (Instant) $\leq n$ Let $\acute{\emptyset}$ is a combined set of CPU with τ & $m \leq n$ & $c_{a1}, c_{b1}, c_{c1}, \ldots. c_m$, capacities $c_i \geq c_{i+1}$ for $\forall$i $= 1, \leq i \leq m$, τ can be scheduled on the uniform MPSoC that meets all the deadlines $u_n \leq c_m$; where $u_a \leq c_a$ for all $k = 1,2,.$ m; Overall utilization factor: $u_{i=}\frac{\alpha_i}{p_i}$ for $\forall$ $c = \tau_i$ (c_i , $p_{i,}$, d_i) and *n* periodic task $\tau = (\tau_1, \tau_2, \tau_3, \tau_4, \tau_5 \ldots., \tau_n)$ scheduled on a CPU the total u_i is defined as:

$$\mathrm{u}_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} \leq 1 \quad (4.2)$$

For $\forall$ load a set of $\tau = \tau_i(c_a, p_{a,}, d_a)$; for $\forall$ *n* periodic task $\{\tau_1, \tau_2, \tau_3 \ldots. \tau_n\}$ with random deadlines scheduled on a CPU, *Load* (τ) = *max* $\{\mathrm{U}_\tau\}$.

The system model is useful for attaining better performance and reliability we are considering an octa-core model for this research. The proposed scheduling technique calculates the utilization of task that starts execution and separate them into a configuration of two, three and 8-cores based on their u_i. The proposed scheduling algorithm is suitable for multi-core and single-core MPSoC platforms.

A maximum allowable u_i value is set for all the cores of the processor by considering different states of the processor running, idle, sleep, and transition from sleep to idle, idle state to running state and sleep to running, The researcher defines a configuration (λ_n)

depending on the u_i of the core. When the task load is based on u_i of the processor reaches near to threshold, maximum allowed u_i for a core to be in running state. The tasks start migrating to the core that is physically distant from the core with high u_i for attaining better performance and avoiding delay in the execution process the configuration (λ_n) selected for the tasks that reach the threshold and need to migrate from sleep mode and prior shifted to idle mode.

These physically distant cores are arranged in the same configuration because of lower utilization u_i effects e.g. Core1 and core7 are far from each other as compared to core1 and core2 because they are very close to each other. So in such a manner cores are selected in a configuration to reduce energy consumption and thermal effects. The proposed scheduler can choose the configuration λ_n based on u_i and task load. Thermal cycles decrease when the number of processors in the running state is decreasing in a configuration due to this idle interval also decreased.

4.3.1 Proposed System Model

The system model mentioned below in Figure 4.1 gives a comprehensive description of the elements that will be involved in conducting experiments. The major components include a task producer, simulator, Core configurations and scheduling algorithm, where the user can create random task sets in xml.txt file that contains different criteria and parameters that include $(a_{i,}\ c_i, p_i, d_{i,} p_r)$. These parameters in the XML file are used as input for the simulation tool e.g., activation time, worst case execution time (WCET), period, deadline, priority, Best case execution time (BCET), and task set generator containing random task sets under different constraints according to the no of cores set by the user in XML.

The utilization is represented as U and N denotes the total no of the task in the ready task set. The proposed scheduling algorithm works for all real-time jobs in a running state using the system model represented in Figure 4.1 allowing all new tasks in the ready state to the scheduling phase and execute on time. The proposed EA-EDF algorithm the scheduling technique improves the chip's overall working performance. When the tasks are ready to run the proposed scheduler verifies that the tasks are compatible with the scheduler and maps to cores based on the utilization factor (u_i). In the proposed system model dynamic fixed priorities are assigned to the entire n periodic task $\tau = \{\tau_{a1}, \tau_{b1}, \tau_{c1} \dots \dots . \pi_n\}$ with deadline d_a is equal to p_a.

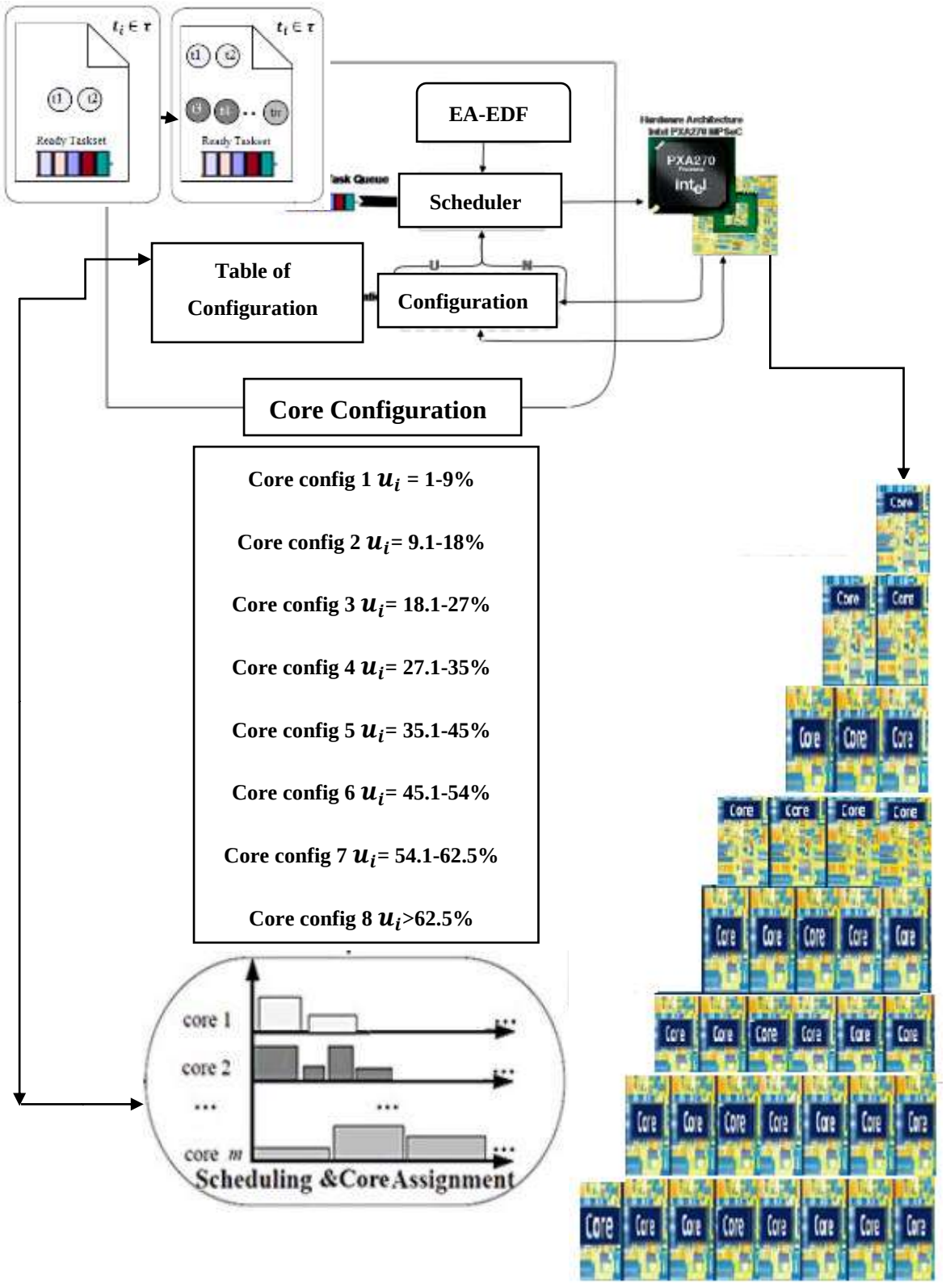


Figure 4. 1 *Functional Overview of the proposed MPSoC System Model*

The proposed system model is based on a comprehensive task scheduling mechanism based on Dynamic power management (DPM) using task migration to reduce the energy

consumption of the MPSoC. The state of the core switching mechanism is based on DPM that switches that sleep state to idle and idle to run if required. The proposed method also uses the full task migration policy when the utilization reaches its threshold to migrate the task to the core that is not in use. DPM moves the remaining cores that are not in use to the sleep state to ensure a significant reduction in energy and power consumption and guarantees the scheduling of tasks by meeting their deadlines.

A simulator STORM simulation tool for real-time multiprocessor scheduling can have a set of ready tasks and simulate the input file according to the scheduling policy by considering task migration and task balancing according to the configuration policy for cores that are developed in the proposed algorithm produces the power profiles of the individual core according to the given set of parameters in XML. The Input XML file can have all the information about the complete simulation duration for the execution of experiments, hardware resources that are required according to the load, architecture for task set and scheduling algorithm. Simulations are considered using OCTA CORE INTEL PXA-270 (LEAT) multiprocessor which is preferable because the processor has an intelligent power-saving mechanism. INTEL PXA-270 (LEAT) multiprocessor can consume power according to the states there are various states run/idle/sleep. In the running state, the processor consumed some active power when the tasks are executing. 925mW power is consumed during running states. The processor behaves idle for all the remaining cores that are not running are supposed to have idle power. In idle state few parameters are in operating because the system clock is performing some jobs but they can be immobilizing. 260mW power is consumed during an idle state. The switching time essential from sleep/standby mode to move to run state is almost the same. When the processor is in sleep/standby mode then there are no peripherals that consume power due to operating and the system clock is also not functional so the power consumption is very low. 1.7224mW power is consumed during standby mode. 0.1630mW power is consumed during sleep mode and 0.1014mW power is consumed during deep sleep mode.

4.3.2 Dynamic Power Management Implementation for EA-EDF Scheduling

DPM is used with various developmental policies of MPSoC in modern embedded system. These policies analyze the response of the system in run time to reduce the overall energy and dissipation of power in MPSoC. During the running state of an application, a selective shift to deep sleep state of the CPU occur that increases the

performance. In embedded systems normally all the system components work in different operating modes because different modes spend different amounts of energy and take different times to return to their normal mode and reduce the total CPU power usage. DPM uses multiple CPU modes sleep (*idle*) and active (running) to efficiently reduce the dissipation of energy. It is feasible to improve total system energy consumption by wisely utilizing these states including low-power state when the system is inactive and turning down the resources during their periods of inactivity. In the proposed system model INTEL PXA270 CPU is considered for the implementation of low power states mainly known as DPM mode. The proposed EA-EDF multiprocessor scheduling algorithm considers the DPM policy to reduce the consumption of energy and takes actions based on dynamic activities in the CPU. The CPU can turn off certain sections mainly the Idle cores of the CPU to a sleep state represented by $P_{sleep} = \beta$ in the proposed system model when the CPU $\alpha = 1$, $\beta = 7$, λ_1=1 when $Core_n$ is executing the instructions ($P_{exe} = \alpha$) the utilization is under the provided limit in the core configurations at u_i=1- 9%, then the rest of the seven cores maintain their sleep state. The objective of using DPM in this research is to switch the m cores of Intel PXA270 MPSoC $P = \{p_1, p_2, \dots, p_m\}$ to a power mode that consumes less energy when the cores are idle and return to full power mode when the periodic task $t_i \in \tau$ $\tau = (t_1, t_2, \dots., t_n)$ arrive and requires to be scheduled.

The state of the core switching mechanism in this research is based on DPM that switches that sleep state to idle and idle to run if required. Table 4.1 represents various power consumption states of Intel PXA-270, PXA250 and Transmeta Crusoe by considering various states of CPU sleep represented as $P_{sleep} = \alpha$, running denoted as $P_{exe} = \beta$ and the idle state as $P_{idle} = \Upsilon$. **Table 4. 1** Power consumption of various MPSoC using DPM

MPSoC	Parameter	Value	MPSoC	Parameter	Value
PXA-270	α	550mW	PXA-250	α	180µw
PXA-270	β	925 mW	PXA-250	β	280 mW
PXA-270	Υ	260 mW	PXA-250	Υ	555 mW
Transmeta Crusoe	α	80µw	Transmeta Crusoe	β	276 mW

At time $T_1 > 0$ is the time between two continuous task arrivals that is represented by $(s_{0,}s_1, \ldots, s_m)$ where as $s_o \in p_m$. ā is an action of task migration that occurs at s_1. Modeling of discrete-time and various MPSoC states P_{sleep}, P_{exe} , P_{idle} and the arrival request occurs due to free memory.

The proposed EA-EDF scheduling based on the DPM switches context state of the CPU processor to low-power state inactive state for as long as possible until all variable tasks completes by meeting their deadlines and utilization u_i of the task, load reaches its threshold. Therefore, a significant amount of reduction in energy dissipation is observed. Suppose the probability (p_e) distribution function is $f(t_1, s_{1,}ā_1)$ of an incident $I = \{t_1, t_2, \ldots, t_n\}$, as a set of collection of periodic task τ arrive before the end of the task $t_i \in \tau$ whereas $\tau = (t_1, t_2, t_3, t_4, t_5 \ldots., t_n)$ when the action ā is selected in the MPSoC system having state $s_{1,}$ having m-processor $P = [s_{1,}s_2, \ldots, s_m]$.

$P(s_{1+1}|s_{1,}ā_1, t_1)$ represents the probability (p_e) of the proposed EA-EDF system when the transition of state occurs at that time the state of the processor is considered as s_{1+1} when the action ā is selected in the MPSoC system having state $s_{1,}$ having m-processor $P = [s_{1,}s_2, \ldots, s_m]$. $K(j_1, s_{1,}ā_{1,})$ is the probability (p_e) that the system transition to state j occurs when action $ā_1$ for migration of task is chosen in the system having state s_1. $Y(s_{1,}ā_1)$ is the expected duration of time in the CPU state s_1 when the action ā is selected in the MPSoC system having state $s_{1,}$ having m-processor $p_m = [s_{1,}s_2, \ldots, s_m]$.

Therefore:

$$K(j_1, s_{1,}ā_{1,}) \int_0^n P(j_1, t_1, s_{1,}ā_{1,}) \,.\, f(dt|s_{1,}ā_1) \tag{4.3}$$

$$Y(s_{1,}ā_{1,})\, t \sum_{s_1 \in ā_{1,}}^{n} P(j_1, t_1, s_{1,}ā_{1,}) \,.\, f(dt|s_{1,}ā_1) \tag{4.4}$$

The energy dissipation and performance degradation cost of $(s_{1,}ā_{1,})$ occurs when an action $ā_1$of the task, switching is taken at the state s_1.

$$E_{cost}(s_{1,}ā_{1,}) = k(s_{1,}ā_{1,}) + \int_o^n f(dt|s_{1,}ā_1) \tag{4.5}$$

$$\sum_{s_{n+1} \in p_{1,}}^{n} \int_1^u R(s_{n+1}, s_{1,}ā_{1,}) \,.\, L(s_{n+1}, |t_1, s_{1,}ā_{1,}) \tag{4.6}$$

When an action $\bar{a}_1$ of a task, switching occurs at the state s_1 the fixed cost $E_{cost}K(s_{1,}\bar{a}_{1,})\ and\ L(\ s_{n+1},|t_1,s_{1,}\bar{a}_1)$ represents the reduced cost of energys_{n+1}.

For solving the constraint of energy optimization and increase in performance the solution to the proposed scheduling policy is defined below in Equation 4.7

$$min \sum_{s_n \in P_,}^{n} \sum_{\bar{a} \neq A_,} \text{Energy}_{\text{consumption}}(s_{1,}\bar{a}_{1,}), f(s_{1,}\bar{a}_{1,}) \quad (4.7)$$

The above Equation 4.7 show that the probability of ensuring minimum energy consumption by switching the states of CPU when there is no action from ā on the core of CPU is not executing any task under $(s_{1,}\bar{a}_{1,})$ the EA-EDF reduces the consumption of energy and also increases the life span of the chip.

DPM is an effective method for the optimization of energy and to achieve high performance we have implemented it using core configurations and a task migration policy followed by the utilization factor u_i of the task load τ: when the m core of Intel PXA270 MPSoC $P = \{p_1, p_2, \dots, p_m\}$ component is not working, it will switch the processor to an inactive state its state into the idle a lower energy state to reduce the consumption of energy. Also decrease the power consumption of the chip by turning off all the cores that are in an idle state.

This technique doesn't consider spatial gradients and reduces average power utilization and temperature on the chip by keeping the idle cores to be turned off when they are not in use.

Table 4.2 elaborates the flow chart of the CPU's transition using DPM in case the state of CPU is $P_{running}$ then selection of relevant suitable configuration occurs and continue normal execution without making any change in the configuration while on other hand in case the state of the CPU is idle then select the suitable configuration and starts transition from processor in P_{idle} to processor in P_{run} for proper execution and scheduling. The no of running processor in greater than the accumulative utilization factor then the Transition of unused idle processor occurs from P_{idle}- P_{sleep} that results in reduction of energy and power consumption

Table 4. 2 *Implementation of DPM for EA-EDF*

Algorithm 1: ***DPM Implementation for EA-EDF***

Input: p_{all}, p_{sleep}, $p_{running}$, *ready* τall, τ *size task parameters, energy and power* (*E&P*) *profiles, state of task, m denotes processor*

τ *ready—Ready task Set*

u_i*—Utilization factor*

τ *size —Size of the task set, Pall— Set of total no of CPU, Psleep—CPU to be in sleep mode*

Generation of Tasks set $t_i \in \tau$ ($r_i, c_i, d_{i,} p_{i,} p_r$) *of Tasks* $t_i \in \tau$

τ*activation*

While EA-EDF

if *(m state* P_{run} *is running in the Selected Configuration) then continue normal execution No change in configuration*

if

(m state is idle in the selected configuration) ***then***

(Transition of state P_{idle}- P_{run})

else if

$P_{run} > u_i$ *then*

Transition of of unused idle state P_{idle}- P_{sleep}

Reduced E&P Consumption

end

return τ *.Finish(),* CPU *sleep, E&P optimized*

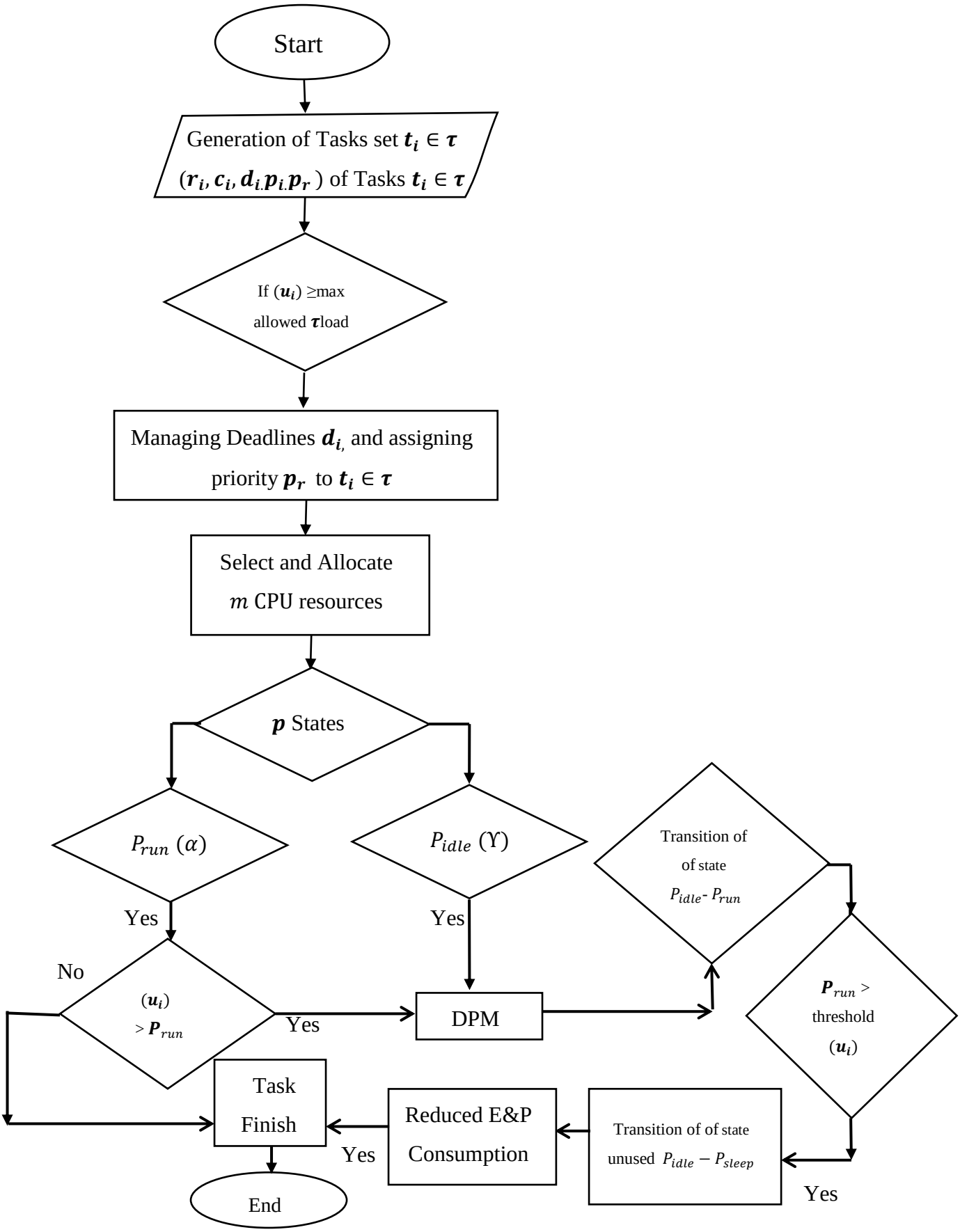


Figure 4. 2 *Flow chart for CPU State transition using DPM*

4.3.3 Energy/Power Model for EA-EDF Scheduling

This section elaborates on the power mode we consider that multiprocessor with m cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ cores in an MPSoC can support DPM. The energy model calculates the dynamic consumption and static power in CMOS circuits. Equation 4.8 illustrates the dynamic power consumption $P_{dynamic}$ of the core in the CMOS circuit on V_{dd}, f voltage frequency level represented by:

$$P_{dyna} = C_{eff}.{V_{dd}}^2.f \tag{4.8}$$

Where V_{dd} is the required supply voltage, while the frequency component is represented as f while C_{eff} shows the switching capacitance V_{ths} is the threshold voltage and R is a constant [190]. The length of the cycle is denoted as t_{cycle} is given as:

$$t_{cycle} = \frac{D_L * R_5}{(V_{dd} - V_{ths})} \tag{4.9}$$

DPM reduces the consumption of dynamic power elements. Sub-threshold leakage is the major contributor to leakage that can enhance significantly with adaptive body biasing.

$$V_{th} = V_{th1} - R_1 * V_{dd} - R_2 * V_{bb} \tag{4.10}$$

The threshold voltage ($V_{th)}$, sub-threshold current (I_{sub-th}) that occurs due to the contribution of static power consumption and cycle time t_{cycle} are considered a function of the V_{dd} and V_{bb}. WhereR_1, $V_1 - R_2$ and V_{th1} are technology constants. The processor power model follows a suitable core configuration. Specifically, the C-states contain one active and rest as low power state (sleep) and deep sleep states. The p-states contain active states and mainly differ in power, energy consumption and operating frequency. In the MPSoC circuit, the number of devices is denoted as n_d Sub-threshold leakage (I_{subl}) contains (I_{rb}) as reverse bias junction current is expressed below in Equation 4.11.

$$P_{leakage} = n_d(v_{dd}I_{subl} + |V_{bs}|I_{rb}) \tag{4.11}$$

Whereas I_{subl} is measured using

$$I_{subl} = R_3(e^{R4*V_{dd}})(e^{R5*V_{bb}}) \quad (4.12)$$

Equation 4.13 represents the consumption of energy over a period $[t_a, t_b]$ can be calculated using:

$$E = \int_{t_a}^{t_b} p(t).dt\,, \quad (4.13)$$

The consumption of energy over workload w_i that requires to be scheduled on a processor with a speed s_a is integral as shown in Equation 4.14.

$$E\,(W,S) = (V.I_{le} + C_a.V^2.f\,).(w_i/s_a) \quad (4.14)$$

Energy consumption of cores of the MPSoC is represented as static power and dynamic power. As long as the CPU is in on state static power is consumed while dynamic power is consumed during computation times as shown in Equation 4.15.

$$\int_0^{tmax} p(t).dt\,, P_{total} = P_{static} + P_{dynamic} \quad (4.15)$$

The authors in [191] proposed a technique that calculates an inherent power P_{on} that is required for keeping the processor in on state, is related to idle power the total power dissipation given by:

$$P_{total} = P_{on} + P_{static} + P_{dynamic} \quad (4.16)$$

$P_{dynamic}$ depicts the various parameters like charging and discharging of the output capacitance while the chip is working and is concerned to the switching α and c_a load capacitance *V* represents the voltage and *f* defines frequency as mentioned below in Eq. (4.17).

$$P_{dynamic} = C_a.V^2.f \quad (4.17)$$

Static power was approximately negligible in earlier processors: For *n* no of transistors the parameter that is considered design-dependent is represented as K_{design}, while I_{leak} is defined as technology-dependent $P_{dynamic}$ parameter.

The dynamic energy per cycle E_d and the leakage energy per cycle is defined in Equation 4.18 and Equation 4.19 respectively.

$$E_{dynamic} = C_{eff}.V^2.f \tag{4.18}$$

$$E_{Per-Cycle} = f^{-1} * n_d * v_{dd} I_{subl} + |V_{bb}| I_{rb} \tag{4.19}$$

Whereas the delay/cycle f^{-1}. Energy needs to keep the system in an ON state per cycle is represented as $E_{on} = f^{-1} * P_{on}$ increases with an increase in lower frequencies are equal to the total energy consumption per cycle as defined in Equation 4.20.

$$E_{total\ Per-Cycle} = E_{dynamic} + E_{Per-Cycle} + E_{on} \tag{4.20}$$

Let's consider switching the overhead of the multiprocessor architecture with m cores with $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, \dots, t_n)$ of the multiprocessor architecture with m cores indicates the duration of minimum time in which the CPU $P = \{p_1, p_2, \dots, p_m\}$ should stay in a sleep state. At a specific time interval when the CPU is considering sleep state that duration is smaller than BET when the state of the processor switches for energy optimization therefore BET can be defined as follows:

DPM is a design technology that reconfigures the whole computing system dynamically in such a way that requested services can be provided with the minimum number of active CPUs with suitable performance levels.

$$TB_e = \max(\delta x \frac{E_x - \delta x_s * P_{\sigma x}}{P_u - P_{\delta x}}) \tag{4.21}$$

The authors in [192] introduce break even time as shown in Equation 4.21 represents the parameter $\mathbf{B_e}$ known as the break-even time. $\mathbf{B_e}$ refers to the minimum duration of idle period that is provided to schedule and utilizes the state named as sleep state $\boldsymbol{\delta x}$ efficiently. $\boldsymbol{B_e}$ is the sum of the total required time to finish the state transition and the duration of idle time that is required to find a way to reduce the shifting energy. The below Equation 4.22 determines the $\boldsymbol{BET_{sleep}}$ break-even time.

$$BET_{sleep} = \max(t_{pa} \frac{E_{Pa-} P_{sleep} * t_{pa}}{P_{idle} - P_{sleep}}) \tag{4.22}$$

$P_{idle}\ and\ P_{sleep}$ shows the idle and sleep power and $P_{idle} > P_{sleep}$

The short circuit power is utilized when both PMOS and NMOS are on for a short period. Equation 4.23 represents the transition of the state $\boldsymbol{E_0}$ and its power dissipation in running and the idle state as $\boldsymbol{P_w}$ and $\boldsymbol{P_s}$.

$$P_w = T_{be} = E_0 + P_s * (T_{be} - T_0) \tag{4.23}$$

In Equation 4.24 (T_{be}) breakeven time measures the length of the idle state of the CPU to optimize power.

$$T_{be} = \left(\frac{E_0 - P_s * T_0}{P_a - P_s}\right) \tag{4.24}$$

During the running state of an application, a selective shut-down of the system components occur that are in the idle state increases the performance. When the CPU starts to transition the accumulative energy required for the transition of state from sleep to idle and from idle to running is represented as E_i and its power dissipation at state 1 is denoted as P_a and at state 2 its P_s. For high-performance delay due to the state, the transition must be less than T_{be} as shown in Equation 4.25:

$$T_{be} = max\left[\left(\frac{E_i - P_s * T_0}{P_a - P_s}\right) * T_0\right] \tag{4.25}$$

$E_t(s)$ is the total energy consumption represented as:

$$E_t(s) = E_{exe}(s) + E_{idl}(s) + E_{sleep}(s) + E_{overhead}(s) \tag{4.26}$$

Whereas $E_{exe}(s)$ is the consumption of energy when the task $t_i$is executing on m core of the processor while $E_{idl}(s)$ is the consumption of energy when the core m of the multiprocessor is in idle mode $E_{sleep}(s)$ represents the sleep state of the core. $E_{overhead}(s)$ is the energy consumption on m core of the processor due to the overhead of state transition. When $E_{idl}(s) < TB_e$ in such a situation the core will be in an idle state and the consumption of energy on the core will be higher due to this the core will not enter into a sleep state.

$$E_{idl}(s) = P_{idle}.t_{idle} \tag{4.27}$$

The energy consumed during E_{sleep}, when $t_{sleep} > P_{sleep}$ is calculated using:

$$E_{sleep} = P_{sleep}.t_{sleep} \tag{4.28}$$

The major goal of proposed energy-aware EDF and DPM is to reduce energy consumption by considering the CPU switching.

Generally, in MPSoC consumption of energy gives a higher hierarchical view of energy saving so it is obvious to measure the energy. However, the impact of the EA-EDF scheduling algorithm and the workload is investigated on different components of the

hardware mainly the CPU because it's a major energy-consuming component. The energy consumption of the CPU decreases with an increase in sampling periods of a system with tasks $\tau = \{\tau_a, \tau_b, \tau_c \ldots\ldots \pi_n\}$ by reducing the workload that results in energy optimization when energy-aware EDF is employed on the energy consumption function of CPU h_1 and h_2 as shown below in Figure 4.4.

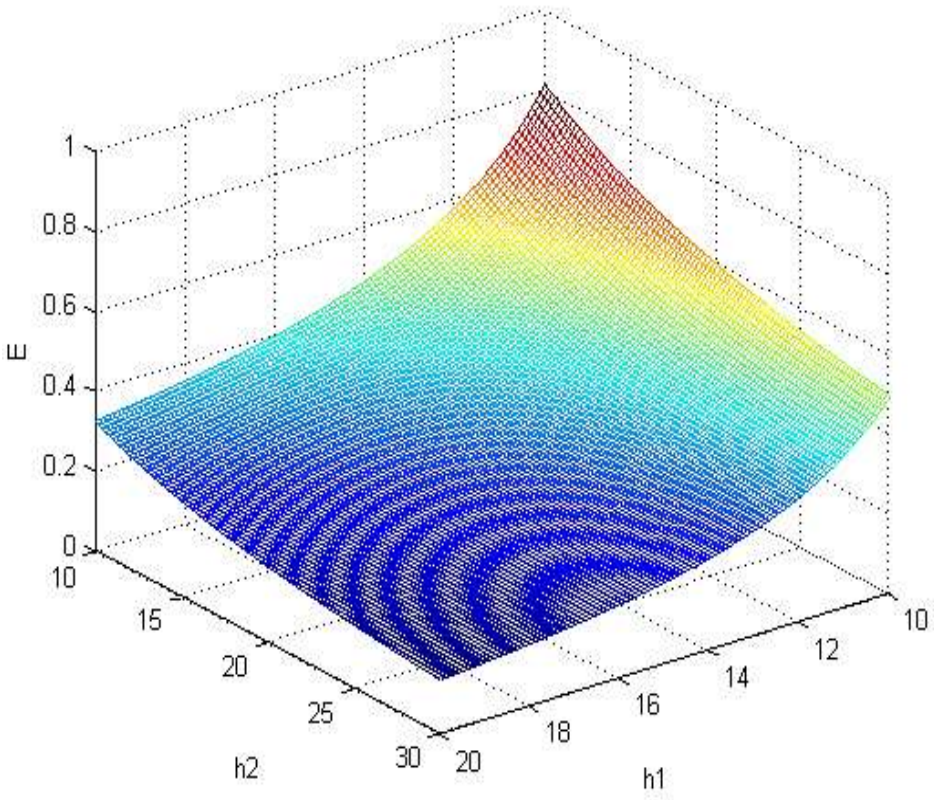


Figure 4. 3 *Normalized CPU energy consumption using the proposed EA.EDF Scheduling Model*

4.3.4 Proposed EA-EDF Scheduling Algorithm

For selection core $\lambda \in \lambda_n$ the counter is used in the algorithm to calculate how many times a configuration is selected that helps to avoid the repetitions of the same configuration λ_n again and again. When a core λ is not in use its counter value is zero for the entire configuration. If the number of the counter is minimum then the configuration can attain the highest priority of repetition.

Once a $\lambda \in \lambda_n$ is selected its counter for the running duration increases unless it reaches the threshold utilization and tasks start migrated to other λ_n with least u_i. The below section elaborates the scheduling algorithm and the flow chart.

Table 4. 3 *Proposed EA-EDF Algorithm with Task Migration*

Proposed Algorithm 2: EA-EDF Algorithm with Task Migration
Input: *P_{all}, P_{sleep}, τ_{all}, τ size Task mappings, Energy and power profiles, migration of task,* *τ_{size} —Size of the task set, Pall— Set of total no of CPU, Psleep—CPU to be in sleep mode* ***∂****— Configurations of Core selection,* *L1—Threshold max_ allowed load of processor, L2—Max load on running core,* ***$L_{current}$— Measure the current load of the processor according to the Utilization factor*** *τ_{ready}—Ready task Set* *C_i —Counter 1= Count time for core selection,* *u_i—Utilization factor* *$Config_k$ —Avalible_core configurations in Dual Core System* *Config $_{RC\ \partial}$ —Core configurations in running state* *Config ∂ — Selection of next core processor* ***For*** *$\forall\tau_{ready} \in P_{all} - P_{sleep}$* ***if*** *(Selected Configuration with Max_ u_i < threshold μ_i)* ***$L_{current}$ < L_2; then*** *No change in configuration_Continue the normal execution* ***else if*** *$L_{current}$ = L_2; (Max_ μ_i configuration==Max_task_allowed)* *No change in P selection_Continue the normal execution* ***For*** *$\forall\tau \in \tau_{all}$* ***do*** *$\tau$ = get utilization factor u_i for all τ_{ready}* ***if*** *Config ∂ (selection of new_Core config ≤ Number_core config_select)* ***then*** *Transition of state from P_{sleep} to P_{idle}* ***else if*** *$L_{current}$ (Current threhold_task load of_Config== L_1 Max threshold _load on_running core)* *Switching of τ to _ Config ∂;* *$Config_{RC}$ ∂ (no of core config executing= Config ∂ (Next processor_ config)* ***end if*** ***until*** *τ is not Ø* *return $CPU_{selected}$, τ migration*

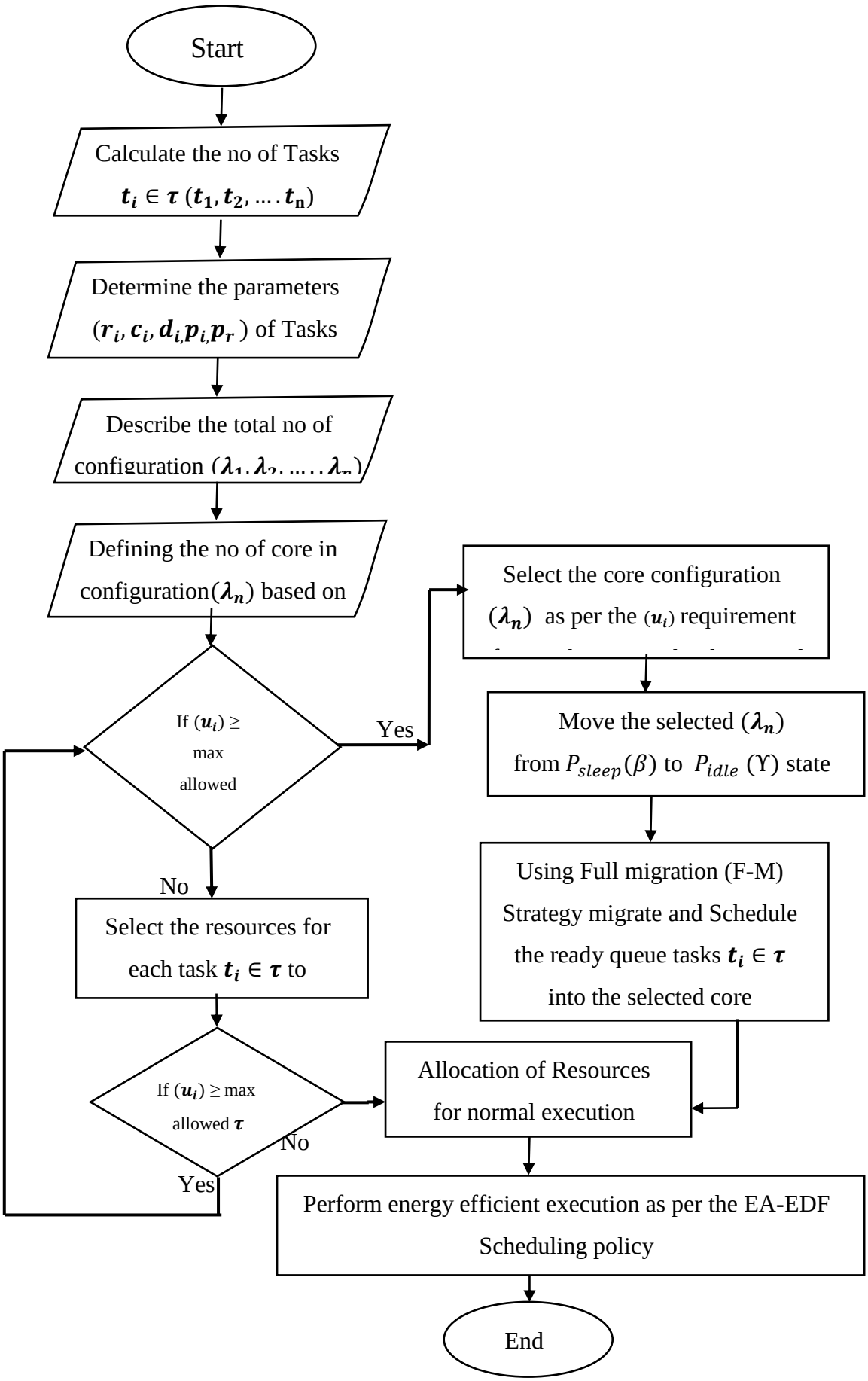


Figure 4. 4 *Flowchart of execution for the proposed energy-aware scheduling*

Figure 4.5 illustrates various steps operations and decisions that are required to execute a process in a sequence on the scheduler in the proposed technique for the octa-core INTEL PXA-270 (LEAT) multiprocessor. The flow chart consists of 12 steps the first step is to divide the applications into tasks and calculate the no of Tasks $t_i \in \tau$ $(t_1, t_2, \dots, t_8)$ as video decoder application consists of 8 periodic tasks. The second step is to determine the parameters $(r_i, c_i, d_{i,} p_{i,} p_r)$ of tasks $t_i \in \tau$ i.e start time, worst-case execution time, period, deadline and priority. The third and fourth step determines the total no of configuration(λ_n), $(\lambda_1, \lambda_2, \dots \dots \lambda_n)$ and defining the no of core in configuration(λ_n) based on (u_i) which requires a core reaching to threshold τ load value. Calculate the utilization factor based on parameters $r_i, c_i, d_{i,} p_{i,} p_r$ of the task set.

If $(u_i) \geq$ max allowed threshold τ load value if we keep using that core serious thermal effects arise which includes thermal hotspots or permanent deformation of chip, in that case, select the core configuration (λ_n) as per the (u_i) requirement of arrived $t_i \in \tau$, either least used (λ_n). Move the selected (λ_n) from $P_{sleep}(\beta)$ to P_{idle} (Υ) state to avoid delays and threshold workload value which is already set to reduce delays and migration cost and if a core reaches a threshold value will shift the workload to that core which is in less use are in sleep mode using full migration (F-M) strategy that migrates and schedules the ready queue tasks $t_i \in \tau$ into the selected core configuration (λ_n) without missing the (d_i) and we will save energy as well as reduce power consumption. In the next step, a CPU is allocated to perform energy-efficient execution as per the EA-EDF Scheduling policy.

If $(u_i) <$ max allowed threshold τ load value then select the resources for each task $t_i \in \tau$ to perform normal execution but with the arrival of new $t_2, t_3 \in \tau$ then calculate the (u_i) requirement of arrived $t_i \in \tau$ if $(u_i) \geq$ max allowed τload then start the above process move the selected (λ_n) from $P_{sleep}(\beta)$ to P_{idle} (Υ) state and if a core reaches a threshold value will shift the workload to that core that is in less use are in sleep mode using full migration (F-M) strategy that migrates and schedule the ready queue tasks $t_i \in \tau$ into the selected core configuration (λ_n) without missing the (d_i). Once the workload is migrated to the core that has the least load the scheduler starts performing the execution of the job. We have considered the configuration of the core (λ_n) based on the utilization factor (u_i). In each configuration, we will consider those cores which are away from each other to reduce thermal gradients effects and save energy as well as reduce the power consumption using the proposed EA-EDF scheduling policy in which

the system with multi-core architecture keeps the cores in a running mode according to utilization factor(u_i) and remaining cores must be placed in sleep mode to optimize energy. `

4.3.5 Proposed Core Configurations ($\boldsymbol{\lambda_n}$)

In proposed EA-EDF scheduling, core selection and switching of τ jobs on various cores configurations are based on the utilization factor (u_i). All the hardware and software parameters are used from Marvell's X-Scale technology based on the Intel PXA270 multiprocessor. The amount of tasks in the running state requires a suitable task migration policy that defines the criteria for scheduling of $ready\ \tau$ on the core. Selection of core depends on various tasks that are in running mode when tasks are in running state the utilization factor remains constant but due to an increase in the number of tasks the utilization factor of the multi-core processor increases as discussed in the system and task model. When tasks are running the u_i remains constant but as the number of tasks increases due to the concurrent processing of tasks on the multi-core processor u_i rapidly increases. For reducing energy and power consumption optimal energy-efficient core configurations are defined for proper core switching and load balancing using task migration based on this policy.

The processor in a configuration that is running is represented as a_i having some operating frequency f_i and the acceptable task load to avoid maximum u_i value known as the threshold value is represented as T_i. The octa-core processor can execute eight tasks at a time each task on an individual core if the processor is executing some task is represented as $P_{exe} = \alpha\ while\ P_{sleep} = \beta$ denotes when the processor is in a sleep state. $P_{exe} = \alpha\ while\ P_{sleep} = \beta$. Core configurations are represented as λ_k few configurations are used for the execution of jobs based on utilization factor as follow. The configurations of *m* cores are defined for the various ranges of utilization u_i using energy-aware EDF scheduling algorithm is mentioned below:

4.3.5.1 Configuration ($\boldsymbol{\lambda_1}$) for Utilization Factor $\boldsymbol{u_i}$ 1% - 9%

When the u_i is from 1% to 9% then 1 core is in on state and seven cores are in sleep mode and there will be only one configuration in which no core is shared

$$\forall \tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} \geq (1 \rightarrow 9\%) \tag{4.29}$$

1cores *1Configuration=1

u_i= 1% - 9%, $\alpha = 1, \beta = 7, \lambda_1$=1

λ_1: (Core1) or (Core2) or (Core3) or (Core4) or (Core5) or (Core6) or (Core7) or (Core8).

One core is in running state and the rest of the seven cores are in sleep mode, the tasks can schedule either one core configuration {(1), (2), (3), (4),(5),(6),(7),(8)}

4.3.5.2 Configuration (λ_2) for Utilization Factor u_i 9.1%-18%

Therefore two cores are in running state and the rest of the six cores remain in sleep mode, tasks can schedule on any of the two core configurations {(1,2),(3,4),(5,6),(7,8)}

$$\forall\tau \to u_\tau = \sum_{k=2}^{n} \frac{c_a}{p_a} \geq (9.1 \to 18\%) \tag{4.30}$$

When the u_i is from 9.1%to18% then 2 cores are in on state and six cores are in sleep mode and there will be only one configuration in which no core is shared.

2cores *1Configuration=2

u_i= 9.1% - 18%, $\alpha = 2, \beta = 6, \lambda_2$=1

Shared Cores=0

λ_2: (Core1, Core2).

4.3.5.3 Configuration (λ_3) for Utilization Factor u_i 18.1%-27%

Therefore three cores are in running state and the rest of the five cores remain in sleep mode, tasks can schedule on any of the core configurations {(1,2,3,),(4,5,6,),(1,7,8)}

$$\forall\tau \to u_\tau = \sum_{k=3}^{n} \frac{c_a}{p_a} \geq (18.1 \to 27\%) \tag{4.31}$$

When the u_i is from 18%to27% then 3 cores are in on state and five cores are in sleep mode then there will be two configurations in which four cores are shared.

3cores *1Configuration=3

u_i= 18.1% - 27% , $\alpha = 3, \beta = 5, \lambda_3$=1

λ_3: Core1, Core2, Core3.

4.3.5.4 Configuration (λ_4) for Utilization Factor u_i 27.1%-36%

Therefore four cores are in running state and the rest of the four cores remain in sleep mode, tasks can schedule on any of the configurations of core set {(1, 2, 3, 4), (5,6,7,8)}

$$\forall\tau \rightarrow u_\tau = \sum_{k=4}^{n} \frac{c_a}{p_a} \geq (27.1 \rightarrow 36\%) \tag{4.32}$$

When the u_i is from 27%to36% then 4 cores are in running state and four cores are in sleep mode then there will be a total of two configurations used and no core is shared.

4cores *1Configuration=4

u_i= 24% - 36%, $\alpha = 4, \beta = 4$, λ_4=1

Shared Cores=0

λ_4: (Core1, Core3, Core 5, Core 7).

4.3.5.5 Configuration $(\boldsymbol{\lambda_5})$ for Utilization Factor $\boldsymbol{u_i}$ 36.1.%-45%

Therefore five cores are in running state and the rest of the three cores remain in sleep mode, tasks can schedule on any of the core configurations {(1, 2, 3,4,5), (4,5,6,7,8) }

$$\forall\tau \rightarrow u_\tau = \sum_{k=5}^{n} \frac{c_a}{p_a} \geq (36.1 \rightarrow 45\%) \tag{4.33}$$

When the u_i is from $(36.1 \rightarrow 45\%)$then 5 cores are in on state and three cores are in sleep mode then there will be a total of two configurations used in which four cores are shared.

5cores *2Configuration=10

u_i= 36% - 45%, $\alpha = 5, \beta = 3$, λ_5=2

Shared Cores=2

λ_5: (Core1, Core3, Core 5, Core 7, Core 8).

λ_5: (Core2, Core4, Core 6, Core 8, Core 1).

4.3.5.6 Configuration $(\boldsymbol{\lambda_6})$ for Utilization Factor $\boldsymbol{u_i}$ 45.1%-54%

Therefore six cores are running and the rest of the two cores remain in sleep mode.

$$\forall\tau \rightarrow u_\tau = \sum_{k=6}^{n} \frac{c_a}{p_a} \geq (45.1 \rightarrow 54\%) \tag{4.34}$$

When the u_i is from 45.1%to54% then 6 cores are in on state and two cores are in sleep mode then there will be a total of two configurations used in which five cores are shared.

6cores *2Configuration=12

u_i= 45% - 54%, $\alpha = 6, \beta = 2$, λ_6=1, shared cores=5

λ_6: (Core1, Core3, Core 4, Core 5, Core 7, Core 8).

λ_6: (Core1, Core2, Core 3, Core 4, Core 5, Core 7).

4.3.5.7 Configuration ($\boldsymbol{\lambda_7}$) for Utilization Factor $\boldsymbol{u_i}$ 54.1%-62.5%

When the u_i is from 54.1% to 62.5% then 7 cores are in on state and two cores are in sleep mode then there will be a total of two configurations used in which five cores are shared. Therefore seven cores are running and the rest of the two cores have remained in sleep mode.

$$\forall\tau \rightarrow u_\tau = \sum_{k=7}^{n} \frac{c_a}{p_a} \geq (54.1 \rightarrow 62.5\%) \tag{4.35}$$

7cores *2configuration=14

u_i= 54.1% to 63%, $\alpha = 7, \beta = 1$, λ_7=1, Shared Cores=6

λ_7: (Core1, Core2, Core3, Core 4, Core 5, Core 6, Core 7)

λ_7: (Core1, Core2, Core 3, Core 4, Core 5, Core 7, Core 8)

4.3.5.8 Configuration ($\boldsymbol{\lambda_8}$) for Utilization Factor $\boldsymbol{u_i}$ >62.5%

When the utilization factor is greater than 62.5% then all 8 cores are in on state.

$$\forall\tau \rightarrow u_\tau = \sum_{k=8}^{n} \frac{c_a}{p_a} > 62.5\% \tag{4.36}$$

8cores *1Configuration=8, shared cores=0

u_i= > 62.5%, $\alpha = 8, \beta = 0$, λ_8=1

λ_8: (Core1, Core2, Core3, Core 4, Core 5, Core 6, Core 7, Core 8).

The authors in [193] proposed demand bound (d_{bi}) is applied with response time to ensure that cores are enough $\forall\tau$ to meet their deadlines (d_a). The demand bound function (DB) gives computation time for $\forall\tau$ to complete the operation within a time interval $[t_a, t_b]$ as shown in Equation 4.37.

$$d_{bi}[t_a, t_b] = c_i \; ; \forall[r_i \geq t_a, \quad d_a \leq t_b] \tag{4.37}$$

Equation 4.38 represents the demand bound for n periodic task set $\tau = \{\tau_a, \tau_b, \tau_c \ldots\ldots \pi_n\}$ with a constrained deadline ($d_a = p_a$).

$$d_{bi}[t_a, t_b] = \sum_{i=1}^{n} db_i[t_a, t_b] \tag{4.38}$$

$$\forall[r_i \geq t_a, \quad d_a \leq t_b] \tag{4.39}$$

4.3.6 Proposed Task Set

Considering s_d=1000ms using proposed EA.EDF scheduling technique contains the hardware and software architecture of the INTEL PXA270 multiprocessor having synthetic task set. p_m represents the no of CPU in the hardware architecture of the octa-core Intel PXA270 MPSoC with a total no of 8 tightly coupled CPU cores that are represented as multiprocessor architecture with m cores with $P = \{p_1, p_2, p_3, p_4, \ldots., p_8\}$ having a total of 2 tasks $t_i \in \tau$ are required for u_i=6% Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2)$ and its parameters for t_1, t_2, t_3, t_4 as a set of periodic task τ, over $m_{(P)}$ processors are as follows:

Table 4. 4*Parameter of task set n = 2*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
t_1	6	0	10	12	1
t_2	7	0	11	12	1

Intel PXA-270 processor's power model is used for each core in the test scenarios because the Intel-PXA270 offers seven power states. We tested the behavior of our proposed method by running all tasks τ till the arrival of the worst-case execution time (WCET). For this, we created a task set and stored the data in an XML input file. The task set's size ranges from 5 to14 tasks with each task's period falling within [0.15ms, 14ms] as indicated. We executed the task set mentioned in Table 4.5 on the Intel-PXA270 MPSoC processing platforms. The XML input contains, starts time, WCET, period, priority, deadline, hardware architecture, number of cores and proposed EAEDF scheduling mechanism. The proposed scheduling algorithm using task migration ability is implemented on homogeneous multi-core architecture that has more than one core and shares the same architecture and microarchitecture integrated on a single chip.

Table 4. 5 *Parameter of task set n = 14*

Tasks	τ1	τ2	τ3	τ4	τ5	τ6	τ7	τ8	τ9	τ10	τ11	τ12	τ13
Period	0.1	9	6	13	9	6	8	10	7	10	12	13	14
Deadline	10	9	6	8	9	8	6	7	6	7	10	5	11
WCET	5	3	2	6	5	4	5	4	5	4	5	4	4
Priority	1	2	3	4	5	4	4	3	7	3	10	11	5
Arrival	0	0	0	2	3	4	2	3	0	3	0	5	6

4.4 Evaluation Requirements

This section describes the complete evaluation requirements for the proposed framework using the proposed EA-EDF Algorithm.

4.4.1 INTEL PXA-270 (LEAT) Multiprocessor

Simulations were completed using INTEL PXA-270 (LEAT) multiprocessor which is preferable because the processor has an intelligent power-saving mechanism. PXA-270 processor can consume power according to the states there are various states run/idle/sleep. In the running state, the processor consumed some active power when the tasks are executing. 925mW power is consumed during running states. The processor behaves idle for all the remaining cores that are not running are supposed to have idle power. In idle state few parameters are in operating because the system clock is performing some jobs but they can be immobilized. 260 mW power is consumed during an idle state. The switching time essential from sleep/standby mode to move to run state is almost the same. When the processor is in sleep/standby mode then there are no peripherals that consume power due to operating and the system clock is also not functional so the power consumption is very low. 0.001724W power is consumed during standby mode. 0.00163w power is consumed during sleep mode and 0.001014W power is consumed during deep sleep mode. Table 4.6 Power consumption specification of Intel PXA-270 MPSoC at various frequencies.

Table 4. 6 *Power consumption specification of Intel PXA-270 MPSoC at various frequencies [193]*

Frequency	Active Power	System Bus	Idle Power	Current
624 MHz	925 mW	208 MHz	260 mW	770 mA
520 MHz	747 mW	208 MHz	222 mW	630 mA
416 MHz	570 mW	208 MHz	186 mW	500 mA
312 MHz	390 mW	208 MHz	154 mW	380 mA
208 MHz	279 mW	208 MHz	129 mW	150 mA
104 MHz	116 mW	104 MHz	64 mW	110 mA

4.4.2 Integration of Proposed EA-EDF Models in multiprocessor scheduling simulation tool

For estimations energy at the system level requires the integration of EA-EDF in the energy/power estimation simulator to achieve the estimation and measurement of the overhead of energy in embedded MPSoC. In this research, the energy targets the design of MPSoC-based embedded systems including hardware components as well as a software components. Energy-aware EA-EDF model is developed in chapter 4 and will be integrated into a simulation tool.

4.4.3 STORM

To simulate the execution state of an application and implementation of proposed policy (EA-EDF) for proper execution using task migration to calculate the energy overhead using dynamic power management (DPM) based low power scheduling policies, we use a simulation tool for real-time multiprocessor scheduling (STORM) is software that is based on java [194]. STORM implements various scheduling techniques on MPSoC-based architecture on both homogeneous and heterogeneous CPUs. The main functionality of the STORM simulator is used to generate the energy consumption profiles, and evaluating of performance of hardware and application platforms. STORM runs on various portable devices and is very flexible for the reason that users can program and insert new components to this (freeware software under creative commons license) java-based software by using a well-defined structured API(S). The simulator contains the architecture of embedded multicore processor design i.e. (MARVEL INTEL PXA-270, 250) as well as the memory architecture. This simulator supports various low energy/power consumption techniques, particularly dynamic power management (DPM) and dynamic voltage scaling (DVFS) and executes any fixed input parameters of the application or either executes input parameters provided by the user at any specific instant of time. This java based application is used to execute the proposed scheduling algorithm for all the cores every single time instant. Figure 4.5 illustrates the platform of the simulation tool. The parameters for hardware and software architecture are simulated through the XML file. The simulation of configuration parameters and scheduling policy is also provided in an XML file so the user can easily evaluate the response of efficiency, tasks, timing and performance through simulation results that are saved in files are either monitored through diagrams. Figure 4.6 illustrates the overview of the STORM simulator and operating procedure.

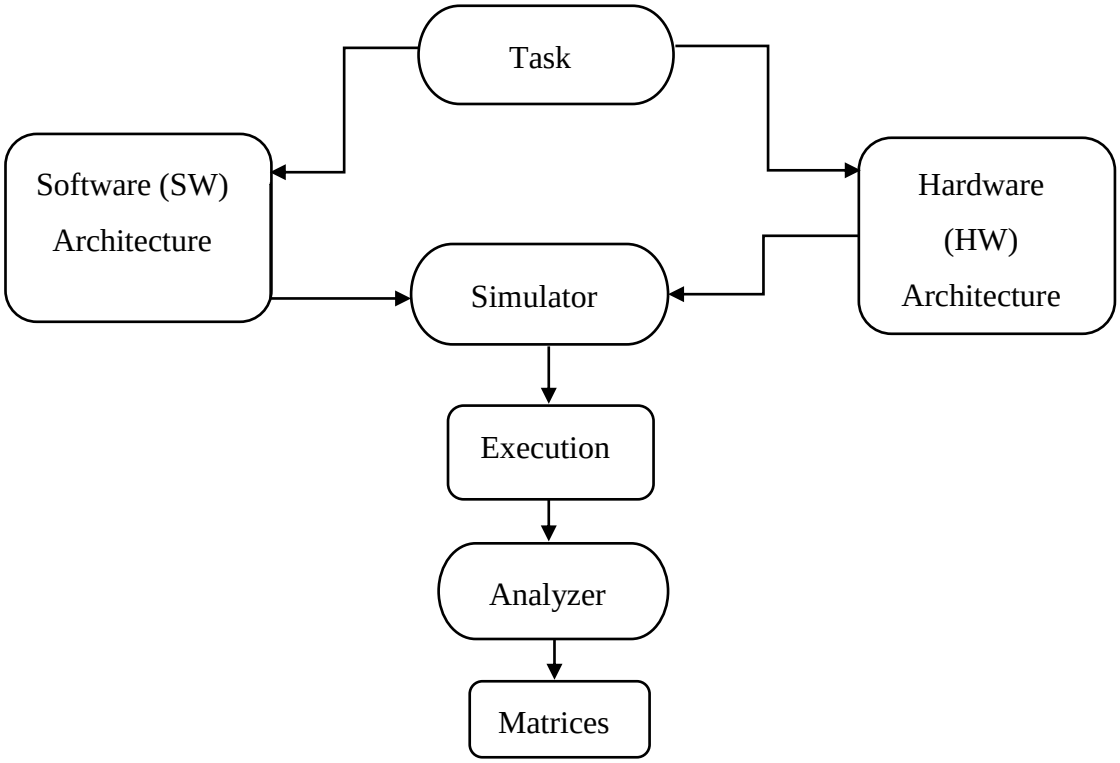


Figure 4. 5 *Simulation platform for real-time scheduling of multi-processor* [194]

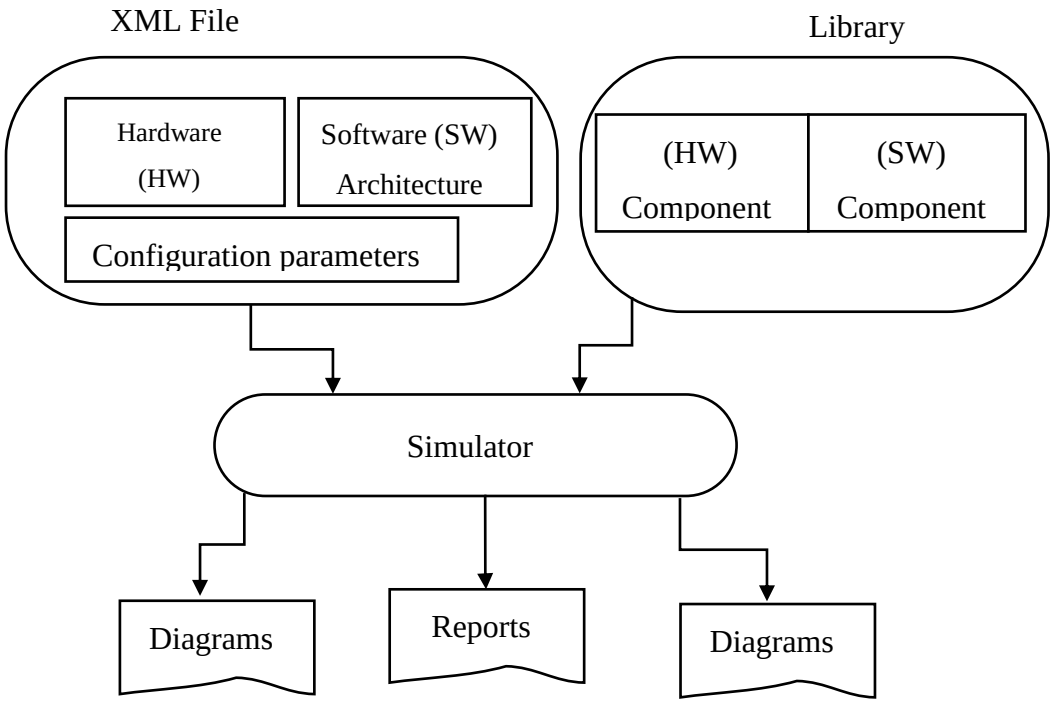


Figure 4. 6 *Overview of STORM simulators input/output file system and operating procedure* [194]

4.4.4 The Architecture of the STORM simulator

The architecture of STORM is based on three major entities put together about a simulation kernel.

4.4.4.1 Software Entity

Tasks and data are software entities for which the simulation is performed. The set of tasks is represented as τ_1, τ_2, τ_3 and D_1, D_2represents the data for m processor.

4.4.4.2 Hardware Entity

Tasks and data are software entities for which the simulation is performed. The set of tasks is represented as task set τ for the software architecture of the task $(t_1, t_2, t_3, t_4, t_5, t_6 \ldots t_n)$ and D_1, D_2 represents the data for the processor $P = \{p_1, p_2, \ldots . p_m]$ are the system's hardware entities.

4.5 System Timing Requirements using proposed EA-EDF Scheduler

The measurements of parameters for hardware and software architecture are simulated through an XML file in the STORM. For $1\% \geq u_i > 62.5\%$ considering the duration feature <SIMULATION duration="1000"> parameters and precision measured in milliseconds for the task set τ for the software architecture of the task $(t_1, t_2, t_3, t_4, t_5, t_6 \ldots t_n)$ for various configurations proposed in section 4.34. s_d represents the duration of simulation as s_d=1000ms for a functional execution in milliseconds. <SCHED> attribute illustrates the proposed (EA-EDF) full preemptive scheduler that is used for simulation is represented as:

<SCHEDclassName="EA.EDF" quantum="10"></SCHED>.

The proposed model contains the hardware and software architecture of the INTEL PXA270 multiprocessor. (p_m) represents the no of CPU in the hardware architecture of the octa-core Intel PXA270 MPSoC with a total no of 8 tightly coupled CPU cores that are represented as multiprocessor architecture with m cores with $P = \{p_1, p_2, \ldots p_6, p_7, p_8\}$ having tasks $t_i \in \tau$. The CPUs part gives information about hardware architecture. The CPU attribute contains a specific Id that is used to represent the simulation kernel and a className used to consider the appropriate library for the proposed system model: Each <TASK> set entity $t_i \in \tau$ is a set of periodic tasks (τ), with no activation memory, no abort on missed deadlines and is represented as $\tau = (t_1, t_2, \ldots t_n)$ over $m_{(P)}$ processors. Ids are assigned to all 8 core that contains individual

tasks and relevant parameter information. Each task $t_i \in \tau$ is linked with the scheduling requirements $(r_i, c_i, d_{i,} p_{i,} and\ p_r)$. The activation date, release, or best case execution time (BCET) of the task r_i represents the time at which the $t_i \in \tau, (t_1, t_2, t_3, t_4, t_5, t_6 \ldots t_n)$ is in the ready queue, while p_i is the period and c_i is the worst-case execution time (WCET), p_r represents the priority of execution for each τ, the deadline is represented as d_i.

DATAS represents a set of task instance that are sent through software from one another

```
<SIMULATION duration="1000">
<SCHED className="EA.EDF"></SCHED>
className=Ć INTEL PXA270MPSoC= LEATProcessor//

<CPUS>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core1" id="1" </CPU>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core2" id="2" </CPU>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core3" id="3" </CPU>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core4" id="4" </CPU>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core5" id="5" </CPU>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core7" id="7" </CPU>

<CPU Ć="storm.Processors.LEATProcessor" name ="CPU Core8" id="8" </CPU>

</CPUS>

<TASKS>

<TASK Ć ="storm.Tasks.PTask_NAM" name ="PTASK T1" id="9" period="10"
activationDate="0" deadline="10" WCET="1"  priority="1"></TASK>s

<TASK Ć ="storm.Tasks.PTask_NAM" name ="PTASK T2" id="10" period="9"
activationDate="2" deadline="4" WCET="1"  priority="6"></TASK>

<TASK Ć ="storm.Tasks.PTask_NAM" name ="PTASK T3" id="11" period="4"
activationDate="4" deadline="20" WCET="1"  priority="7"></TASK>

</TASKS> </SIMULATION>
```

Figure 4. 7 *XML Parameters for System Timing Requirements Input for Proposed EA-EDF Scheduling Algorithm XML Input File*

Task set for system timing requirements (ms), at u_i=6% for task set n=2 m = 1 considering s_d=1000ms using proposed EA.EDF scheduling technique contains the hardware and software architecture of the INTEL PXA270 multiprocessor. p_m represents the no of CPU in the hardware architecture of the octa-core Intel PXA270 MPSoC with a total no of 8 tightly coupled CPU cores that are represented as multiprocessor architecture with m cores with $P = \{p_1, p_2, p_3, \ldots., p_8\}$ having a total of 2 tasks $t_i \in \tau$ are required for u_i=6% can schedule on a core configuration ($\lambda_1 \in \lambda_n$)

having m = 1. Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2)$ and its parameters for (t_1, t_2) as a set of periodic task τ, over $m_{(P)}$ processors as shown in Table 4.7.

Table 4. 7 *Parameters for system timing requirements (ms), at U_i=6% for task set n=2* $m = 1$

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
$\boldsymbol{t_1}$	6	0	10	12	1	$\boldsymbol{t_2}$	7	0	11	12	1

Task set for system timing requirements (ms), at u_i=10% for task set τ n=4 m = 2

considering s_d=1000 ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m *cores of processor* $P = \{p_1, p_2, p_3, p_4, \ldots, p_8\}$ and a total of 4 tasks $t_i \in \tau$ are required for u_i=10% can schedule on a core configuration ($\boldsymbol{\lambda_2 \in \lambda_n}$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, t_3, t_4)$ as a set of periodic task τ, (t_1, t_2, t_3, t_4) over $m_{(P)}$ processors as shown in Table 4.8.

Table 4. 8 *Parameters for system timing requirements (ms), at u_i=10% for task set τ n=4* $m = 2$

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
$\boldsymbol{t_1}$	4	4	4	2	7	$\boldsymbol{t_3}$	16	11	16	2	5
$\boldsymbol{t_2}$	15	10	15	2	8	$\boldsymbol{t_4}$	15	8	15	3	6

Task set for system timing requirements (ms), at U_i=20% for task set τ n=6 m = 3

Considering s_d=1000 ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m cores with $P = \{p_1, p_2, p_3, p_4, \ldots, p_8\}$ and a total of n=6 tasks $t_i \in \tau$ are required for u_i=20% can schedule on a core configuration ($\lambda_3 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, \ldots, t_n)$ and its parameters for $(t_1, t_2, \ldots, t_6)$ as shown in Table 4.9.

Table 4. 9 *Parameters for system timing requirements (ms), at U_i=20% for task set τ n=6 $m = 3$*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
$\boldsymbol{t_1}$	10	0	10	1	1	$\boldsymbol{t_4}$	15	0	40	1	8
$\boldsymbol{t_2}$	9	2	4	2	6	$\boldsymbol{t_5}$	17	2	20	1	5
$\boldsymbol{t_3}$	4	4	20	1	7	$\boldsymbol{t_6}$	22	4	20	2	10

Task set for system timing requirements (ms), at u_i=31% for task set τ n=10 $m = 4$ considering s_d=1000ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m cores with $P = \{p_1, p_2, p_3, p_4, \ldots\ldots, p_8\}$ and a total of n=10 tasks $t_i \in \tau$ are required for u_i=31% can schedule on a core configuration ($\lambda_4 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, \ldots\ldots, t_n)$ and its parameters for $(\mathrm{t}_1, \mathrm{t}_2, \ldots\ldots, \mathrm{t}_{10})$ as shown in Table 4.10.

Table 4. 10 *Parameters for system timing requirements (ms), at u_i=31% for task set τ n=10 $m = 4$*

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

Table 4. 11 *Parameters for system timing requirements (ms), at u_i=38% for task set τ n=13 $m = 5$*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
$\boldsymbol{t_1}$	10	0	10	4	1	$\boldsymbol{t_8}$	4	4	4	2	6
$\boldsymbol{t_2}$	9	0	9	1	2	$\boldsymbol{t_9}$	15	0	15	1	7
$\boldsymbol{t_3}$	6	0	6	2	3	$\boldsymbol{t_{10}}$	19	2	17	1	10
$\boldsymbol{t_4}$	10	0	10	1	4	$\boldsymbol{t_{11}}$	17	4	22	2	1
$\boldsymbol{t_5}$	9	2	9	3	5	$\boldsymbol{t_{12}}$	22	0	23	1	5
$\boldsymbol{t_6}$	6	4	6	2	5	$\boldsymbol{t_{13}}$	25	2	25	1	10
$\boldsymbol{t_7}$	9	2	9	1	1						

Task set for system timing requirements (ms), at u_i=38% for task set τ n=13 $m = 5$ considering s_d=1000ms using proposed EA.EDF scheduling technique for

multiprocessor architecture with m cores with $P = \{p_1, p_2, \ldots, p_8\}$ and a total of n=13 tasks $t_i \in \tau$ are required for u_i=38% can schedule on a core configuration ($\lambda_5 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, t_3, \ldots . t_n)$ and its parameters for $(t_1, t_2, \ldots \ldots, t_{13})$ as shown in Table 4.11.

Task set for system timing requirements (ms), at U_i=50% for task set τ n=17 $m = 6$ considering s_d=1000ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m cores with $P = \{p_1, p_2 \ldots, p_8\}$ and a total of n=17 tasks $t_i \in \tau$ are required for u_i=50% can schedule on a core configuration ($\lambda_6 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, t_3, \ldots . . t_n)$ and its parameters for $(t_1, t_2, \ldots \ldots, t_{17})$ as shown in Table 4.12.

Table 4. 12 *Parameters for system timing requirements (ms), at u_i=50% for task set τ n=17 $m = 6$*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
t_1	10	0	10	4	1	t_{10}	19	2	17	1	10
t_2	9	0	9	3	2	t_{11}	17	4	22	2	1
t_3	6	0	6	2	3	t_{12}	22	0	23	1	5
t_4	10	0	10	5	4	t_{13}	25	2	25	1	10
t_5	9	2	9	3	5	t_{14}	6	4	6	2	5
t_6	6	4	6	2	5	t_{15}	9	2	9	1	1
t_7	9	2	9	3	1	t_{16}	4	4	4	2	6
t_8	4	4	4	2	6	t_{17}	15	0	15	1	7

Task set for system timing requirements (ms), at U_i=55% for task set τ n=19 $m = 7$ considering s_d=1000 ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m cores with $P = \{p_1, p_2, p_3, p_4, \ldots . , p_8\}$ and a total of n=19 tasks $t_i \in \tau$ are required for u_i=55% can schedule on a core configuration ($\lambda_7 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, t_3, \ldots . . t_n)$ and its parameters for $(t_1, t_2, \ldots \ldots, t_{19})$ as shown in Table 4.13.

Table 4. 13 *Parameters for system timing requirements (ms), at U_i=55% for task set τ n=19 $m = 7$*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
t_1	10	0	10	4	1	t_{11}	17	4	22	2	1
t_2	9	0	9	3	2	t_{12}	22	0	23	1	5

t_3	6	0	6	2	3	t_{13}	25	2	25	1	10
t_4	10	0	10	5	4	t_{14}	6	4	6	2	5
t_5	9	2	9	3	5	t_{15}	9	2	9	1	1
t_6	6	4	6	2	5	t_{16}	4	4	4	2	6
t_7	9	2	9	3	1	t_{17}	15	0	15	1	7
t_8	4	4	4	2	6	t_{18}	19	2	17	1	10
t_9	15	0	15	1	7	t_{19}	17	4	22	2	1
t_{10}	19	2	17	1	10						

Task set for system timing requirements (ms), at U_i=62.5% for task set τ n=24 $m = 7$ considering s_d=1000ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m cores with $P = \{p_1, p_2, \ldots\ldots, p_8\}$ and a total of n=24 tasks $t_i \in \tau$ are required for u_i=62.5% can schedule on a core configuration ($\lambda_8 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, \ldots\ldots, t_n)$ and its parameters for $(t_1, t_2, \ldots\ldots, t_{24})$ as shown in Table 4.14.

Table 4. 14 *Parameters for system timing requirements (ms), at u_i=62.5% for task set (τ) n=24 $m = 7$*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
t_1	10	0	10	4	1	t_{13}	25	2	25	1	10
t_2	9	0	9	3	2	t_{14}	6	4	6	2	5
t_3	6	0	6	2	3	t_{15}	9	2	9	1	1
t_4	10	0	10	5	4	t_{16}	4	4	4	2	6
t_5	9	2	9	3	5	t_{17}	15	0	15	1	7
t_6	6	4	6	2	5	t_{18}	19	2	17	1	10
t_7	9	2	9	3	1	t_{19}	17	4	22	2	1
t_8	4	4	4	2	6	t_{20}	15	0	15	1	7
t_9	15	0	15	1	7	t_{21}	19	2	17	1	10
t_{10}	19	2	17	1	10	t_{22}	17	4	22	2	1
t_{11}	17	4	22	2	1	t_{23}	6	4	6	2	5
t_{12}	22	0	23	1	5	t_{24}	9	2	9	1	1

Task set for system timing requirements (ms), at U_i>62.5% for task set τ n=27 $m = 8$ considering s_d=1000ms using proposed EA.EDF scheduling technique for multiprocessor architecture with m cores with $P = \{p_1, p_2, \ldots\ldots, p_8\}$ and a total of n=27 tasks $t_i \in \tau$ are required for u_i>62.5%. Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, \ldots\ldots, t_n)$ and its parameters for $(t_1, t_2, \ldots\ldots, t_{27})$ as shown in Table 4.15.

Table 4. 15 *Parameters for system timing requirements (ms), at u_i>62.5% for task set τ n=27* $m = 8$

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
t_1	10	0	10	4	1	t_{15}	9	2	9	1	1
t_2	9	0	9	3	2	t_{16}	4	4	4	2	6
t_3	6	0	6	2	3	t_{17}	15	0	15	1	7
t_4	10	0	10	5	4	t_{18}	19	2	17	1	10
t_5	9	2	9	3	5	t_{19}	17	4	22	2	1
t_6	6	4	6	2	5	t_{20}	15	0	15	1	7
t_7	9	2	9	3	1	t_{21}	19	2	17	1	10
t_8	4	4	4	2	6	t_{22}	17	4	22	2	1
t_9	15	0	15	1	7	t_{23}	6	4	6	2	5
t_{10}	19	2	17	1	10	t_{24}	9	2	9	1	1
t_{11}	17	4	22	2	1	t_{25}	4	4	4	2	6
t_{12}	22	0	23	1	5	t_{26}	15	0	15	1	7
t_{13}	25	2	25	1	10	t_{27}	19	2	17	1	10
t_{14}	6	4	6	2	5						

4.6 H.264 Multimedia decoder Application

This section describes the implementation of our energy-aware proposed framework represented in the Figure 4.1 using the proposed EA-EDF algorithm on H.264 multimedia decoder application. We have implemented the EA-EDF energy optimization method at the functional level. There are various parameters but the design is explored using the no of processors (CPUs) and operating frequency. The aim of considering this application is to explore the contribution of proposed scheduling mechanism. Moreover, for embedded systems, H.264 decoding application is a promising standard considering high quality, compression, resolution and flexible coding. H.264 multimedia decoder application is a high-quality video compression algorithm consisting of several slices and various frames that are divided into 7 main periodic tasks characterization. Task set consist of various ready task that includes new frames that is derived from the application layer of the network. The application layer decomposes each task into multiple slices as frame 1, and frame 2. The processing task decodes the following slices slice 1(quantization), slice 2(entropy), and slice 3(inverse transformation). Figure 4.8 illustrates the overview of the slice version of the H.264 decoder application introduced by authors in [195] in a frame that contains 3 slices and each slice is further divided into 4 independent subtasks that are executing the frames of the slice in parallel. On the arrival of the new frame, the subtask sequentially accesses the buffer of input data but

frames have temporal dependencies due to this the execution of the newly arrived frame is not possible until the current running frame has completed its execution and properly decoded. So for smooth execution, it is required for the CPU to synchronize the entire task using TASK-SYNC before executing the new frame. Simulation shows that the target architecture consists of uniform processor $P = [p_1, p_2, \ldots, p_m]$ for $t_i \in \tau$ supporting dynamic power management (DPM) mechanism reduces the total energy consumption of the CPU using energy-aware scheduling.

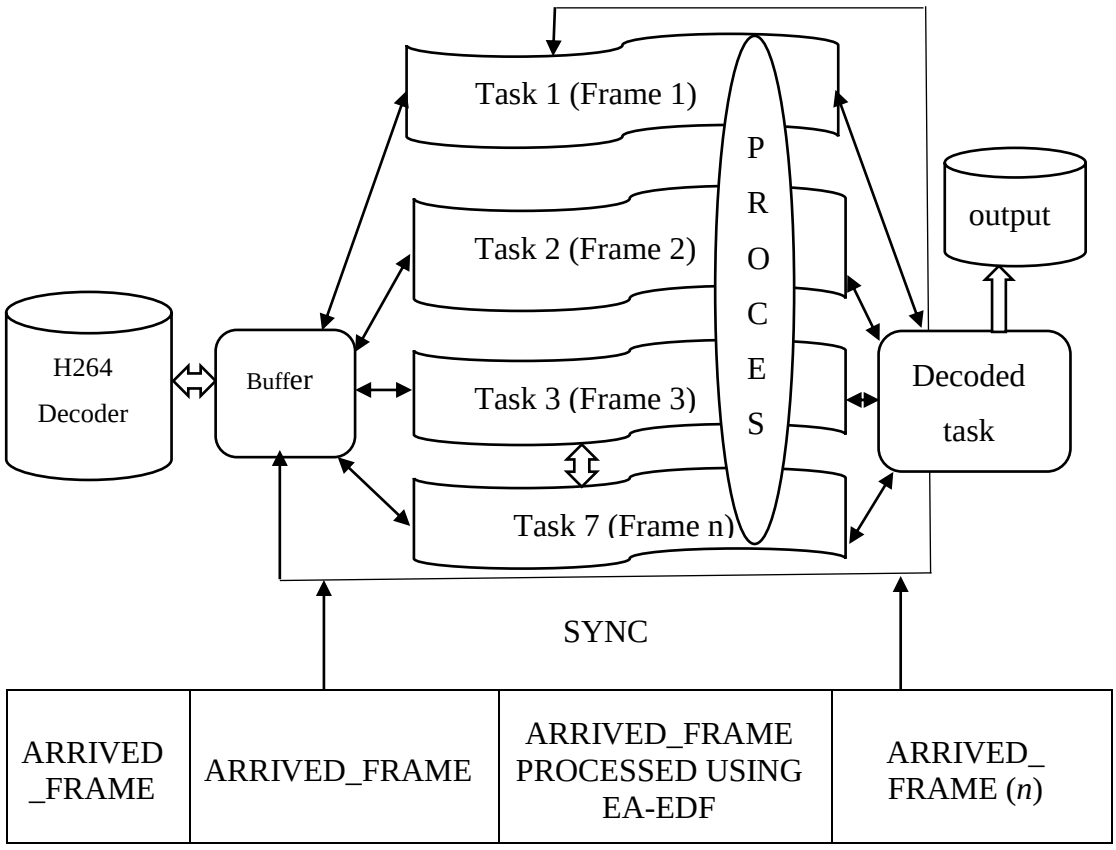


Figure 4. 8 *Overview of Block diagram for H.264 Video decoder slices version* [195]

For a task $\tau = (t_1, t_2, t_3, t_4, t_5, .., t_7)$, $t_i \in \tau$ labeled as frame 1,2,3 is characterized by the following parameters $(a_{i,}, c_i, d_{i,} p_{i,} p_r)$, where the release time or activation period of the task a_i it's the time when the $t_i \in \tau$ scheduling request arrives, represents the time at which the $t_i \in \tau$ is in the ready queue, while c_i is the worst-case execution time, p_i represents the period and the deadline is denoted as d_i. p_r represents the priority. In this simulation, we have considered the slice version of the H.264 video decoder application developed by the authors in [160]. Table 4.9 represents the seven periodic tasks $\tau = (t_1, t_2, t_3, t_4, t_5, .., t_7)$ used in the H.264 application. The parameters of these

task sets are evaluated at the maximum operating frequency of MARVEL INTEL PXA270 MPSoC (i.e., 624-MHz).

Task set for system timing requirements of H.264 is evaluated on simulation duration s_d=1000 ms using proposed EA.EDF scheduling technique for multiprocessor architecture with $m = 8$ cores with $P = \{p_1, p_2, \ldots\ldots, p_8\}$ and a total of n=7 tasks $t_i \in \tau$ required resources to schedule on a core configuration ($\lambda_7 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, t_3, \ldots\ldots t_n)$ and its parameters $a_{i,}, c_i, d_i p_i p_r$ for $(t_1, t_2, \ldots\ldots, t_7)$ are as follows:

Table 4. 16 *H.264 video decoder application parameters for system timing requirements (ms), for the task set τ n=7 $m = 5$ at 624-MHz*

Tasks τ	*Period $p_i(ms)$*	*Activation $a_i(ms)$*	*Deadline d_i*	*Worst case execution $c_i\ (ms)$*	*Priority p_r*
t_1	19	0	19	1	1
t_2	5	0	5	2	1
t_3	66	0	66	42	1
t_4	66	1	66	42	1
t_5	66	2	66	42	1
t_6	66	3	66	42	1
t_7	66	66	66	2	1

The simulation results of the H.264 video decoder application using the proposed EA-EDF scheduling technique are evaluated and discussed in chapter 5, section 5.48.

4.7 Summary

In this chapter, the first section elaborates on the problem statement and contains a detailed overview of the proposed research methodology. A system model is developed for (EA-EDF) scheduling as well as energy and power model for the proposed model is elaborated in detail. A dynamic power management policy that selectively changes the states of the CPU is introduced. Proposed a method for core configurations and various system timing requirements based on the utilization factor is introduced. The proposed algorithm and its flowchart and various components used in the evaluation are adapted to evaluate the proposed energy-efficient scheduling algorithm based on DPM (EA-EDF) to increase the performance of the system by reducing the energy dissipation using suitable abilities of the proposed task migration policy. The next chapters present the

experimental model and simulation results for the proposed DPM based energy-aware efficient task scheduling for MPSoC and illustrated a variety of comparative results on various utilization factor u_i *= 6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* u_i *> 62.5% (63%)* at various operating frequencies 624MHz, 520MHz, 416 MHz, 312 MHz and 208 MHz and compared to previously deployed energy optimization techniques for MPSoCs.

CHAPTER 5

EXPERIMENTAL MODEL AND SIMULATION RESULTS FOR OPTIMAL DPM-BASED EA-EDF

CHAPTER 5

EXPERIMENTAL MODEL & SIMULATION RESULTS FOR OPTIMAL DPM-BASED (EA-EDF) SCHEDULING

5.1 Introduction

This chapter illustrates the experimental techniques and contains a complete explanation of the experimental model mechanism used in the research work.

5.2 Experimental Setup

The proposed EA-EDF algorithm is based on java language compiled in Neon Eclipse that executes the storm.jar file. The proposed algorithms schedule the predefined set of parameters given in XML input files provided to the simulator STORM as input to schedule the task set using EA-EDF. STORM gives energy and power profiles these profiles are then plotted by creating a utilization graph in MatLab that shows the difference between the proposed EA-EDF, and other globally renowned techniques concerning utilization factors. The below figure illustrates the complete experimental setup of the research design that is based on dynamic power management and a comprehensive approach of a full task migration technique that considers an application by dividing it into various no of tasks according to the parameter. Storm can execute input parameters provided by the application using Extensible Markup Language (XML). The features of the task set include the ($a_i, c_i, d_{i,} p_{i,} p_r$) application architecture, start time, worst-case execution time, period, priority, and deadline, hardware architecture no of cores of INTEL PXA-270, simulation duration s_d and proposed EA-EDF scheduling algorithm is added in XML.

Definition: Consider INTEL PXA-270 multiprocessor with 8-cores and the current 2 processors are running the n=3 task under u_i 17% but suddenly with the arrival of a new task n=6, $t_i \in \tau$ the u_i increases to its threshold and suddenly migration of task occurs at u_i=18.1% the scheduler schedule and migrate the $t_i \in \tau$ to the coolest core that is in sleep mode and is prior shifted to idle mode before the migration of the task using the proposed EA.EDF scheduling technique for multiprocessor architecture with m=8 cores $P = \{p_1, p_2, \ldots\ldots, p_8\}$ and a total of n=6 tasks $t_i \in \tau$ are required for u_i=18.1% can schedule on a core configuration ($\lambda_3 \in \lambda_n$). Each task $t_i \in \tau$ consists of a periodic task

set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, \ldots\ldots, t_n)$ and its parameters for $(t_1, t_2 \ldots, t_6)$ as in input to the simulator in Table 5.1

Table 5. 1 *Task set parameters*

Tasks τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r	*Tasks* τ	$p_i(ms)$	$r_i(ms)$	d_i	$c_i\ (ms)$	p_r
t_1	10	0	10	1	1	t_4	6	4	6	2	5
t_2	9	2	4	2	6	t_5	9	2	9	1	1
t_3	4	4	20	1	7	t_6	4	4	4	2	6

The above parameter is imported in the XML file and then given to the simulator using the HW and SW libraries of the multiprocessor on STORM where the energy-aware scheduling policy based on DPM using task migration policy is already implemented on Eclipse JAVA Neon 64bit using the storm jar file the simulator produces various energy and power efficient profiles according to the load and utilization factor (u_i), CPU load curves, CPU memory consumption, CPU energy consumption for individual cores using both hardware & software architecture of the processor model INTEL PXA-270 (LEAT) for all types of hardware configurations already mentioned in the XML file as input to the simulator is executed using multicore processors. The proposed system model asses the reliability using the below-mentioned experimental approach in Figure 5.1.

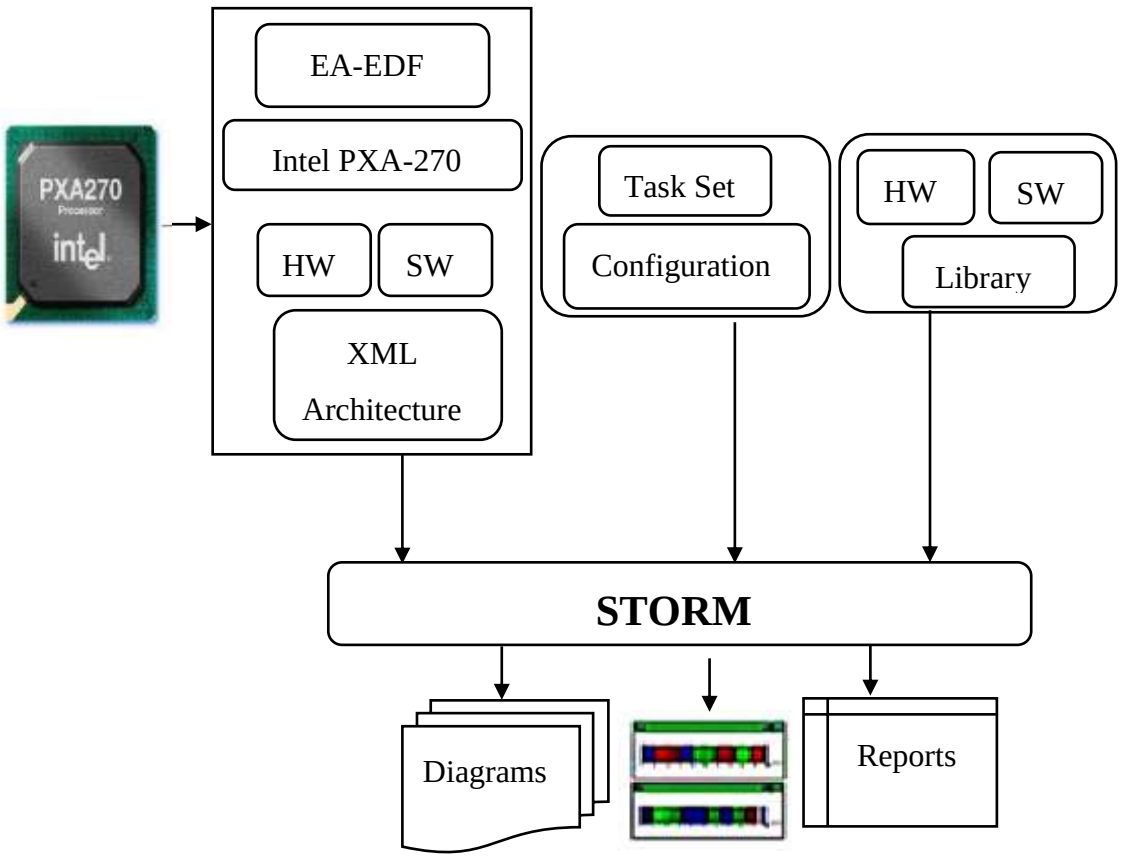


Figure 5. 1 *Experimental Setup for energy-efficient task scheduling* [194]

Table 5. 2 *Sequence of simulation*

Proposed Algorithm	
Step 1:	Calculate and measure the no of tasks $t_i \in \tau$ $(t_1, t_2, \ldots . t_n)$ and adjust the simulation parameters hardware architecture of Marvel X-Scale Intel PXA-270 MPSoC no of cores and application $(r_i, c_i, d_{i,} p_{i,} p_r)$ of the tasks $(t_i \in \tau)$. ,
Step 2:	Generate configurations $(\lambda_1, \lambda_2, \ldots . . \lambda_n)$ and the no of CPUs (cores) in each configuration(λ_n) based on (u_i).
Step 3:	Power Setting of INTEL PXA-270 multiprocessor and its states according to the frequency $P_{sleep}(\beta)$, P_{idle} $(\Upsilon)$and $P_{running}$ (α).
Step 4:	Compute System U_{total} $(\tau) \stackrel{def}{=} \sum_{i=1}^{n} \frac{c_a}{p_a} = u_i$ for the different cores for the workload assessments using Equation 3.2
Step 5:	Compute (u_i) If (u_i) ≥max allowed τ load then adopt the full task migration policy for scheduling and migration of tasks. If(Max u_i configuration < Threhold u_i) then continue the normal execution.
Step 6:	Select the core configuration (λ_n) as per the (u_i) requirement of arrived $t_i \in \tau$, either least used (λ_n) & switch (λ_n) from $P_{sleep}(\beta)$ to P_{idle} (Υ). next_selcted Core config(λ_n) ≤ Core configuration_select than move the CPU from $P_{sleep}(\beta)$to P_{idle}.
Step 7:	Using the full migration (F-M) Strategy migrate and schedule the ready queue of tasks $t_i \in \tau$ into the selected core configuration (λ_n) without missing their deadlines(d_i) using proposed EA-EDF.
Step 8:	Select the $P_{exe}(\alpha)$ core resources in $(\lambda_1, \lambda_2, .. \lambda_n)$ for migratable tasks i.e t_n to least used suitable λ to achieve energy efficient τ execution.
Step 9:	Measurement of the energy-efficient profiles, on various λ_n and $u_i = 6\%, 10\%, 20\%, 31\%, 38\%, 50\%, 55\%, 62.5\%$ and $u_i > 62.5\%)$ as well as reduced power consumption and load plots using the proposed energy-aware EA-EDF algorithm.

The simulation model is useful for attaining better performance and reliability for these various configurations (λ_n) are used that are depending on the u_i of the core. DPM for low power mode and a task migration policy is introduced. When the utilization of the processor reaches near to threshold. The tasks start migrating to the core that is physically distant or to the core configuration (λ_n) that is least used and low consumption and temperature for attaining better performance by avoiding the delay in the execution process. The selection of core configuration (λ_k) is based on the state of the core. Due to high τ load on the CPU tasks $t_i \in \tau$ that reaches the threshold needs to be scheduled and migrated to the coolest core that is in sleep mode and is prior shifted to idle mode before the migration of the task. The proposed scheduling technique calculates the energy and power and switches the τ tasks on the core based on their utilization factor. The u_i of the core is increasing once it is selected for the migrated tasks $t_i \in \tau$ due to this frequent increase in the utilization u_i switching of core maintains the overall execution. When the u_i value of the currently executing core is high than a core (λ_k) with the least task τ load is selected for execution and all the other cores will remain in sleep mode. In this research, the researcher performed experiments on the INTEL PXA-270 multiprocessor labeled as a LEAT processor using the proposed EA.EDF_P_ Scheduler. The above Table 5.2 represents the sequence of simulations used in the experimental technique.

5.3 Simulation Results

To evaluate the performance and functionalities of EA-EDF various findings based on simulations and measurements are conducted using STORM. The STORM simulator can allocate and schedule $t_i \in \tau$ according to various parameters r_i, c_i, d_i, a_i, p_r considered to allow more low consumption states for reducing the consumption of energy and power DPM is applied during idle intervals of the CPU using the proposed EA-EDF scheduler. STORM is used to perform the experimental valuation of a multiprocessor system and illustrate the importance due to advancements in semiconductor technology that increases the power density of a multiprocessor with the increase in demand for R-T applications based on complex circuitry. STORM receives ready tasks $t_i \in \tau$ $(t_1, t_2, \ldots . t_n)$ from the XML input file as shown in the system model proposed in section 4.3.1 and schedules it on the hardware architecture of Marvel X-Scale Intel PXA-270 MPSoC by considering the suitable core configuration λ_n with low-power DPM capabilities. λ_n are chosen based on u_i whereas the power model for all cores is the same using the Intel PXA-270

processor's power model is used for each core in the evaluation because it's one of the prominent embedded MPSoC, Intel-PXA270 offers multiple power states. We tested the behavior of the proposed EA-EDF method by running all tasks τ till the arrival of the worst-case execution time (WCET). For this, we created a task set and stored the parameters data in an XML. The task set's size ranges from [1 to 37] tasks with each task's period falling within [0.15ms, 30ms] as indicated in the task set parameter for various u_i proposed in section 4.5. We executed all the task sets mentioned in section 4.5 using Intel-PXA270 MPSoC processing platforms. Table 5.3 represents various power consumption states of Intel PXA-270 over multiple operating frequencies respectively by considering various states of running, idle, and sleep and also represents the current of the Intel PXA-270 MPSoC. All hardware characteristics from the Intel-PXA270 processor are employed in our experiments using the proposed EA-EDF policy and achieve better performance, reduction in energy power consumption and minimized delays during the execution process. The proposed EA-EDF migrates tasks to a core that is physically away from the core with the highest power consumption and utilization.

5.3.1 Simulation Metrics

The description of the various metrics used for simulation is mentioned in Table 5.3.

Table 5. 3 *Parameters of the experimental system*

Metric system	Values
Simulation Duration s_d	1000 ms
Total no task ($t_i \in \tau$) n	2,4,5,10,13,17,19,24,27
Total no processor m	8
$m_{running}$ duration	c_i, a_i
Failure to miss $d_i$deadlines	No
Task abortion	No
CPU	INTEL PXA-270 MPSoC
u_i	6,10,20,31,38,55,62.5 and >62.5
Scheduler	EA-EDF
Distribution of task's period falling within	[0.15ms, 50ms]

This section presents the experimental evaluation of our improved energy-efficient scheduling algorithm for the reduction of energy consumption. The proposed energy aware-EDF-based scheduler gives improved results and more energy optimization as compared to previous techniques. The Intel PXA-270 MPSoC is used to measure CPU energy usage. According to the proposed algorithm, the job with the earliest scheduling deadline is prioritized When the periodic task set is examined the strategy works well that's why the context switch is valuable. Considering the same experimental characteristics such as deadlines and consumption. The simulation indicates that the STORM simulator performance criteria are equivalent to the real platform values. Below tables represents the comparison of experimental results of energy consumption at various u_i= *6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* $u_i \geq$ *62.5%* when all the cores are running.

5.3.2 Energy Consumption using EA-EDF at 624 MHz, 520 MHz, 416 MHz, 312 MHz, 208MHz, and 104 MHz,

Table 5.4 represents the energy consumption on m cores CPU using the proposed EA-EDF at various workloads u_i= 6%, 10%, 20%, 31%, 45%, 55%, 62.5% and u_i > 62.5% when all the cores are in running condition by considering a multiprocessor with m cores $P = \{p_1, p_2, \ldots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \ldots., t_n)$ in an MPSoC using DPM that allows more low consumption states for reducing the consumption of energy and power. As DPM is applied during idle intervals of the CPU using the proposed EA-EDF scheduler. At 624 MHz Intel PXA270 multiprocessor consumes 0.925 W in active mode and 0.26 W in idle mode. It is also observed from the CPU load curve that high task utilization increases the power densities, energy dissipation and increasing the resistance in MPSoC that causes to reduce the performance as the idle interval is reduced and the running interval is increased.

Table 5. 4 *Energy consumption at different* u_i *on Intel PXA-270 MPSoC at 624 MHz*

Utilization u_i*%*	**Task (τ)**	$Core_{exe}(P)$	$Core_{sleep}(P)$	**Proposed EA-EDF**
u_i *= 6*	2	1	7	0.93 J
u_i *= 10*	4	2	6	0.97 J
u_i *= 20*	14	3	5	1.12 J

u_i = *31*	16	4	4	1.71 J
u_i = *38*	19	5	3	2.86 J
u_i = *50*	24	6	2	3.86 J
u_i = *62.5*	26	7	1	4.19 J
$u_{i>}$ *62.5*	27	8	0	8.60 J

The energy consumption using EA-EDF by considering a multiprocessor with m cores $P = \{p_1, p_2, \ldots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \ldots., t_n)$ in an MPSoC at various workloads u_i= 6%, 10%, 20%, 31%, 45%, 55%, 62.5% and u_i > 62.5% when all the cores are in running condition. As DPM is applied that enhances the state and idle intervals of the CPU by switching using the proposed EA-EDF scheduler. At 520 MHz Intel PXA270 multiprocessor consumes 0.747 W in active mode and 0.222 W in idle mode. The simulation results in Table 5.5 show that the proposed EA-EDF is more energy efficient and reduced energy consumption at 520 MHz when the least no tasks are scheduled due to which less no of processing cores are executing the task and more cores remain in sleep or idle state that causes the processor to consume less energy at u_i= 6%, $\mathrm{Core}_{\mathrm{exe}}(\mathrm{P})$=1 and $\mathrm{Core}_{\mathrm{sleep}}(\mathrm{p})$ =7 consumes very minimal energy while consuming high energy at $u_i > 62.5\%$ $\mathrm{Core}_{\mathrm{exe}}(\mathrm{P})$=8 and $\mathrm{Core}_{\mathrm{sleep}}(\mathrm{p})$ =0.

Table 5. 5 *Energy consumption at different u_i on Intel PXA-270 MPSoC at 520 MHz*

Utilization u_i*%*	**Task (τ)**	$\boldsymbol{Core_{exe}(P)}$	$\boldsymbol{Core_{sleep}(P)}$	**Proposed EA-EDF**
u_i = *6*	2	1	7	0.57 J
u_i = *10*	4	2	6	0.75 J
u_i = *20*	14	3	5	1.02 J
u_i = *31*	16	4	4	2.31 J
u_i = *38*	19	5	3	3.05 J
u_i = *50*	24	6	2	3.89 J
u_i = *62.5*	26	7	1	4.20 J
$u_{i>}$ *62.5*	27	8	0	8.10 J

The above results show a substantial reduction in the energy consumption by lowering

the frequency from 624-520MHz gives greater gain due to increases in the task load the utilization factor increases that increase the no of active processors *m* in running state α simultaneously reduces the idle period Υ that leads to a delay in execution due multiple ready task instructions against limited no of CPU processing capacity. By reducing the frequency a nonlinear decrease in energy consumption ultimately reduces the overall energy dissipation using the EA-EDF scheduler.

The energy consumption using EA-EDF by considering a multiprocessor with m cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ in an MPSoC at various workloads. As DPM is applied that enhances the low power state and idle intervals of the CPU using the proposed EA-EDF scheduler. At 416 MHz Intel PXA270 multiprocessor consumes 0.57 W in active mode and 0.186 W in idle mode. The simulation results in Table 5.6 show that the proposed EA-EDF is more energy efficient and reduced energy consumption at 416 MHz The below results show a substantial reduction in the energy consumption by lowering the frequency and proper scheduling of tasks meeting all the deadlines at 416MHz By reducing the frequency from 520-416MHz a nonlinear decrease in energy consumption occurs ultimately achieving a prominent reduction in the overall energy dissipation using EA-EDF scheduler.

Table 5. 6 *Energy consumption at different u_i on Intel PXA-270 MPSoC at 416 MHz*

Utilization u_i%	**Task (τ)**	$Core_{exe}(P)$	$Core_{sleep}(P)$	**Proposed EA-EDF**
u_i = 6	2	1	7	0.47 J
u_i = 10	4	2	6	0.55 J
u_i = 20	14	3	5	0.98 J
u_i = 31	16	4	4	1.37 J
u_i = 38	19	5	3	2.15 J
u_i = 50	24	6	2	2.80 J
u_i = 62.5	26	7	1	3.20 J
$u_{i>}$ 62.5	27	8	0	7.10 J

The energy consumption using EA-EDF by considering a multiprocessor with m cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ in an MPSoC at various workloads. As DPM is applied that enhances the state and idle intervals of the

CPU by switching of the CPU using the proposed EA-EDF scheduler. At 312 MHz Intel PXA270 multiprocessor consumes 0.39 W in active mode and 0.154 W in idle mode. The simulation results in Table 5.7 show that the proposed EA-EDF is more energy efficient and reduced energy consumption at 312 MHz. The below results show a substantial reduction in the energy consumption by lowering the frequency and proper scheduling of tasks meeting all the deadlines at 312 MHz By reducing the frequency from 416-312 MHz a nonlinear decrease in energy consumption occurs ultimately achieving a prominent reduction in the overall energy dissipation using EA-EDF scheduler.

Table 5. 7 *Energy consumption at different u_i on Intel PXA-270 MPSoC at 312 MHz*

Utilization u_i%	**Task (τ)**	$Core_{exe}(P)$	$Core_{sleep}(P)$	**Proposed EA-EDF**
u_i = 6	2	1	7	0.28 J
u_i = 10	4	2	6	0.45 J
u_i = 20	14	3	5	0.92 J
u_i = 31	16	4	4	1.24 J
u_i = 38	19	5	3	2.15 J
u_i = 50	24	6	2	2.91 J
u_i = 62.5	26	7	1	3.90 J
$u_{i>}$ 62.5	27	8	0	3.10 J

Tables 5.8 and 5.9 represent the energy consumption using EA-EDF by considering a multiprocessor with m cores $P = \{p_1, p_2, \ldots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \ldots., t_n)$ in an MPSoC and supporting DPM at various workloads u_i= *6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* $u_i \geq$ *63%* when all the cores are in running condition It is also observed from the CPU load curve that high task utilization increases the power densities, energy dissipation and increasing the resistance in MPSoC that causes to reduce the performance. As DPM is applied that enhances the low power state and idle intervals of the CPU using the proposed EA-EDF scheduler. At 208 MHz Intel PXA270 multiprocessor consumes 0.51 W in active mode and 0.279 W in idle mode. The simulation results in Tables 5.8 & 5.9 show that the proposed EA-EDF is more

energy efficient and reduced energy consumption at 208 & 104 MHz at 104 MHz Intel PXA270 multiprocessor consumes 0.116 W in active mode and 0.064W in idle mode. The below results show a substantial reduction in the energy consumption by lowering the frequency and proper scheduling of tasks meeting all the deadlines at 208 & 104 MHz by reducing the frequency from 312MHz to 208MHz & from 208MHz to 104 a nonlinear decrease in energy consumption occurs ultimately achieving a prominent reduction in the overall energy dissipation using EA-EDF scheduler.

Table 5. 8 *Energy consumption at different u_i on Intel PXA-270 MPSoC at 208 MHz*

Utilization u_i%	**Task (τ)**	$Core_{exe}(P)$	$Core_{sleep}(P)$	**Proposed EA-EDF**
u_i = 6	2	1	7	0.25 J
u_i = 10	4	2	6	0.35 J
u_i = 20	14	3	5	0.62 J
u_i = 31	16	4	4	0.9 J
u_i = 38	19	5	3	1.55 J
u_i = 50	24	6	2	2.1 J
u_i = 62.5	26	7	1	2.70 J
$u_{i>}$ 62.5	27	8	0	2.95 J

Table 5. 9 *Energy consumption at different u_i on Intel PXA-270 MPSoC at 104 MHz*

Utilization u_i%	**Task (τ)**	$Core_{exe}(P)$	$Core_{sleep}(P)$	**Proposed EA-EDF**
u_i = 6	2	1	7	0.22 J
u_i = 10	4	2	6	0.31 J
u_i = 20	14	3	5	0.6 J
u_i = 31	16	4	4	0.81 J
u_i = 38	19	5	3	1.35 J
u_i = 50	24	6	2	1.92 J
u_i = 62.5	26	7	1	2.40 J

$u_{i>}$ *62.5*	27	8	0	2.65 J

The simulation results in tables 5.8 & 5.9 show that the proposed EA-EDF is more energy efficient and a reduction in energy consumption occurs in Intel PXA-270 MPSoC at 208 and 104 MHz at lower u_i and frequency consumes very minimal energy while consuming high energy at $u_i > 62.5\%$ $\text{Core}_{\text{exe}}(\text{P})$=8 and $\text{Core}_{\text{sleep}}(\text{p})$ =0.

5.3.3 Comparison of Energy Consumption using EA-EDF at 624 MHZ Energy Consumption using EA-EDF at 624 MHz, 520 MHz, 416 MHz, 312 MHz, 208MHz, and 104 MHz,

Tables 5.10 represents the comparison of simulation results conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= *6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* u_i *>62.5 63%* considering the HW parameter of Intel PXA270 multiprocessor with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots., t_n)$ in an MPSoC and supporting DPM and show that the proposed EA-EDF is more energy efficient and prioritized the job with the earliest scheduling deadline and a significant reduction in energy consumption occurs at lower utilization and gives 4.7% more energy efficient results as compared to GEDF, RT-DPM, TBP, PDTM on 624MHz in terms of energy.

Table 5. 10 *Comparative analysis of energy consumption on various schedulers at 624 MHz*

Utilization $u_i\%$	**Task (τ)**	**Proposed EA-EDF**	**Energy GEDF**	**Energy PDTM**	**Energy Uniform RT-DPM**	**Energy TBP**
$u_i = 6$	1	0.93 J	1.41 J	1.18 J	0.86 J	1.25 J
$u_i = 10$	3	0.97 J	1.81 J	2.02 J	1.27 J	1.87 J
$u_i = 20$	15	1.12 J	2.73 J	1.99 J	1.98 J	3.21 J
$u_i = 31$	16	1.71 J	3.2 J	3.91 J	3.52 J	3.79 J
$u_i = 38$	19	2.86 J	6.12 J	4.37 J	4.12 J	4.88 J
$u_i = 50$	24	3.86 J	7.23 J	6.32 J	6.44 J	5.21 J
$u_i = 62.5$	26	4.19 J	8.26 J	6.38 J	5.76 J	6.96 J
$u_{i>}$ *62.5*	28	8.6 J	8.63 J	8.79 J	8.83 J	9.12 J

Tables 5.11 represents the comparison of simulation results conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= *6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* u_i *>62.5%* considering the HW parameter of Intel PXA270 multiprocessor with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ in an MPSoC and supporting DPM and show that the proposed EA-EDF is more energy efficient and prioritized the job with the earliest scheduling deadline and a significant reduction in energy consumption occurs at lower utilization and gives 4.3% more energy efficient results as compared to GEDF and gives much improved results in comparison with RT-DPM, TBP, PDTM on 520MHz in terms of energy at higher u_i.

Table 5. 11 *Comparative analysis of energy consumption on various schedulers at 520 MHz*

Utilization u_i%	**Task (τ)**	**Proposed EA-EDF**	**Energy GEDF**	**Energy PDTM**	**Energy Uniform RT-DPM**	**Energy TBP**
u_i *= 6*	1	0.57 J	2.41 J	0.66 J	2.51 J	2.88 J
u_i *= 10*	3	0.75 J	3.81 J	3.97 J	4.87 J	3.02 J
u_i *= 20*	14	1.02 J	5.73 J	5.41 J	5.21 J	4.77 J
u_i *= 31*	16	2.31 J	6.12 J	6.89 J	6.79 J	6.22 J
u_i *= 38*	19	3.05 J	6.72 J	5.2 J	7.88 J	7.13 J
u_i *= 50*	24	3.89 J	7.23 J	7.44 J	9.21 J	7.72 J
u_i *= 62.5*	26	4.20 J	8.17J	8.76 J	10.96 J	8.38 J
$u_{i>}$ *62.5*	27	8.10 J	8.34 J	9.38 J	11.12 J	9.43 J

Tables 5.12 represents the comparison of simulation results conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= *6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* u_i *> 62.5%* considering the HW parameter of Intel PXA270 multiprocessor with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ in an MPSoC and supporting DPM and show that the proposed EA-EDF is more energy efficient and prioritized the job with the earliest scheduling deadline and a significant reduction in energy consumption occurs at lower utilization and gives 5.3% more energy efficient results as compared to GEDF,RT-DPM

,TBP, PDTM on 416MHz in terms of energy at higher u_i.

Table 5. 12 *Comparative analysis of energy consumption on various schedulers at 416 MHz*

Utilization $u_i\%$	**Task (τ)**	**Proposed EA-EDF**	**Energy GEDF**	**Energy PDTM**	**Energy Uniform RT-DPM**	**Energy TBP**
$u_i = 6$	1	0.47 J	2.34 J	0.16 J	1.51 J	1.88 J
$u_i = 10$	3	0.55 J	3.41 J	2.97 J	2.7 J	2.02 J
$u_i = 20$	14	0.98 J	4.53 J	3.41 J	5.21 J	2.77 J
$u_i = 31$	16	1.37 J	5.2 J	4.89 J	5.79 J	3.22 J
$u_i = 38$	19	2.15 J	6.72 J	4.2 J	5.88 J	3.9 J
$u_i = 50$	24	2.80 J	6.23 J	5.44 J	7.21 J	4.72 J
$u_i = 62.5$	26	3.20 J	7.17J	7.76 J	7.96 J	5.38 J
$u_{i>}$ *62.5*	27	7.10 J	8.34 J	8.38 J	10.12 J	5.43 J

Tables 5.13 represents the comparison of simulation results conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i*= 6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* u_i *> 62.5%.* Considering the HW parameter of Intel PXA270 multiprocessor with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ in an MPSoC and supporting DPM and show that the proposed EA-EDF is more energy efficient and prioritized the job with the earliest scheduling deadline and a significant reduction in energy consumption occurs at lower utilization and gives 5.9% more energy efficient results as compared to GEDF, RT-DPM, TBP, PDTM on 312MHz in terms of energy at higher u_i.

Table 5. 13 *Comparative analysis of energy consumption on various schedulers at 312 MHz*

Utilization $u_i\%$	**Task (τ)**	**Proposed EA-EDF**	**Energy GEDF**	**Energy PDTM**	**Energy Uniform RT-DPM**	**Energy TBP**
$u_i = 6$	1	0.28 J	1.5 J	0.16 J	1.51 J	0.88 J
$u_i = 10$	3	0.45 J	1.61 J	1.97 J	2.47 J	1.12 J

$u_i = 20$	14	0.92 J	3.53 J	2.41 J	2.51 J	1.37 J
$u_i = 31$	16	1.24 J	4.2 J	2.89 J	2.75 J	2.22 J
$u_i = 38$	19	2.15 J	4.72 J	3.2 J	3.5 J	2.4 J
$u_i = 50$	24	2.91 J	5.23 J	7.44 J	4.11 J	3.72 J
$u_i = 62.5$	26	3.90 J	5.77J	7.76 J	5.6 J	4.38 J
$u_{i>}\ 62.5$	27	3.10 J	8.34 J	6.38 J	10.12 J	9.43 J

Tables 5.14 represents the comparison of simulation results conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads considering the HW parameter of Intel PXA270 multiprocessor with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, t_5 \dots, t_n)$ in an MPSoC and supporting DPM and show that the proposed EA-EDF is more energy efficient and prioritized the job with the earliest scheduling deadline and a significant reduction in energy consumption occurs at lower utilization and gives 6.7% more energy efficient results on 208MHz and 7.3% on 104MHz as compared to GEDF, RT-DPM, TBP, PDTM on 208MHz in terms of energy at higher u_i.

Table 5. 14 *Comparative analysis of energy consumption on various schedulers at 208 MHz*

Utilization $u_i\%$	**Task (τ)**	**Proposed EA-EDF**	**Energy GEDF**	**Energy PDTM**	**Energy Uniform RT-DPM**	**Energy TBP**
$u_i = 6$	1	0.25 J	0.35 J	0.16 J	0.51 J	0.88 J
$u_i = 10$	3	0.35 J	1.21 J	1.17 J	1.47 J	2.12 J
$u_i = 20$	14	0.62 J	1.53 J	1.41 J	3.51 J	1.37 J
$u_i = 31$	16	0.9 J	3.2 J	1.89 J	6.75 J	3.22 J
$u_i = 38$	19	1.55 J	2.72 J	3.2 J	5.5 J	3.4 J
$u_i = 50$	24	2.1 J	4.23 J	4.44 J	6.11 J	4.72 J
$u_i = 62.5$	26	2.70 J	4.77J	6.76 J	6.6 J	5.38 J
$u_{i>}\ 62.5$	27	2.95 J	6.34 J	7.38 J	7.12 J	5.43 J

Table 5. 15 *Comparative analysis of energy consumption on various schedulers at 104 MHz*

Utilization $u_i\%$	**Task (τ)**	**Proposed EA-EDF**	**Energy GEDF**	**Energy PDTM**	**Energy Uniform RT-DPM**	**Energy TBP**
$u_i = 6$	1	0.22 J	0.15 J	0.56 J	1.51 J	0.18 J
$u_i = 10$	3	0.31 J	1.21 J	1.57 J	1.7 J	1.12 J
$u_i = 20$	14	0.6 J	3.53 J	1.51 J	2.51 J	2.37 J
$u_i = 31$	16	0.81 J	3.2 J	1.59 J	2.75 J	3.22 J
$u_i = 38$	19	1.35 J	4.72 J	2.2 J	2.95 J	3.54 J
$u_i = 50$	24	1.92 J	5.4 J	3.44 J	4.11 J	4.72 J
$u_i = 62.5$	26	2.40 J	5.57J	4.76 J	4.6 J	4.98 J
$u_{i>}\ 62.5$	27	2.65 J	5.74 J	6.38 J	4.9 J	7.43 J

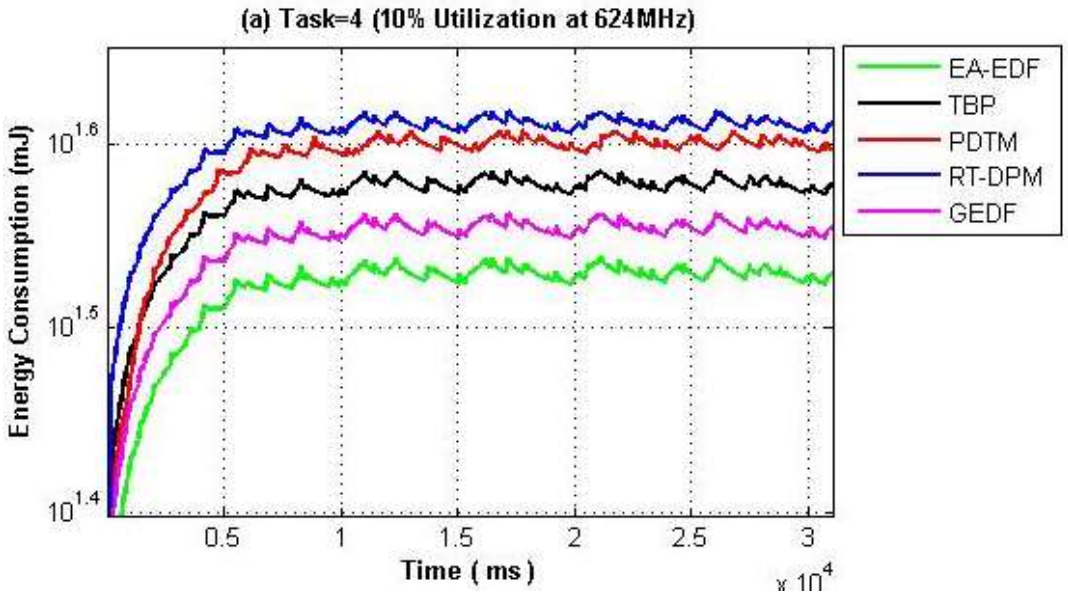


Figure 5. 2 *Comparison of energy consumption u_i= 7%, at 624 MHz*

Figure 5.2 represent the comparison of simulation conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= 7% considering the hardware HW parameter of Intel PXA270 multiprocessor along with the software SW parameters defined in XML after running the simulation STORM gives energy consumption profiles on an individual core that are plotted in the above figure with

m = 8 cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4)$ in an MPSoC.

X-axis is representing the time frame while Y-axis denotes the total dynamic energy consumption of our approach and other benchmarks in joules and supporting. Overall reduction in consumption of energy using EA-EDF occurs when the tasks are properly migrated to the least used core according to the variation in change in u_i utilization. Due to avaibilbilty of low power processor in the core configurations for proper task mapping significant reduction in energy consumption occurs.

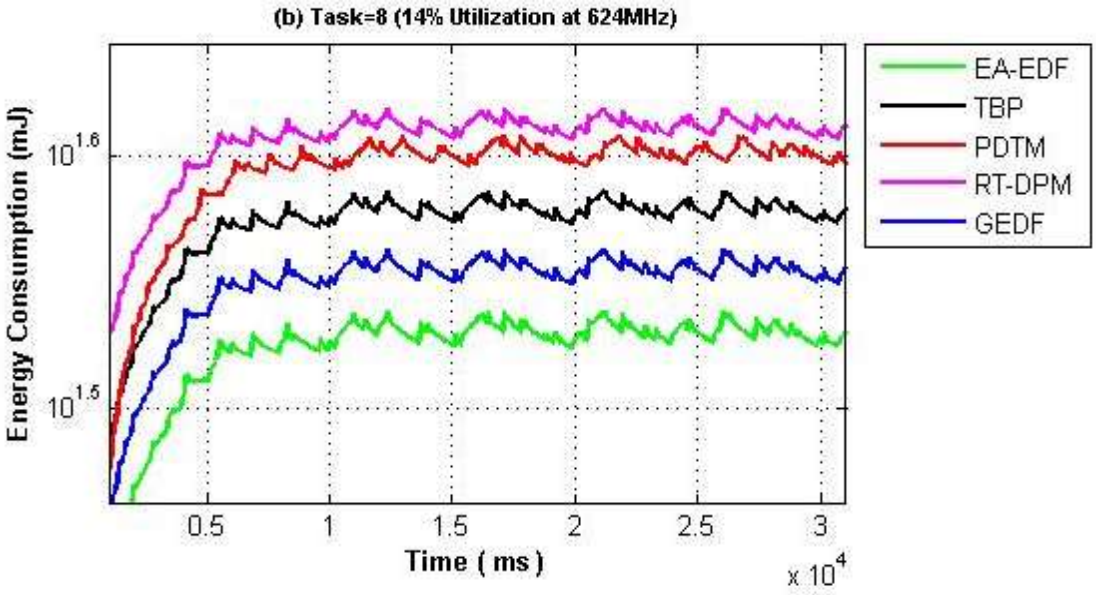


Figure 5. 3 *Comparison of energy consumption u_i= 14% at 624 MHz*

Figure 5.3 represent the comparison of simulation conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= 14% with m = 8 cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, \dots . t_{16})$ total of 3 cores are enough to schedule the task in an MPSoC. X-axis is representing the time frame while Y-axis denotes the total dynamic energy consumption of our approach and other benchmarks in joules and supporting.

Overall reduction in consumption of energy using EA-EDF occurs when the tasks are properly migrated to the least used core according to the variation in change in u_i utilization. Due to avaibilbilty of low power processor in the core configurations for proper task mapping substantial reduction in the energy consumption occurs as compared to other conventional techniques.

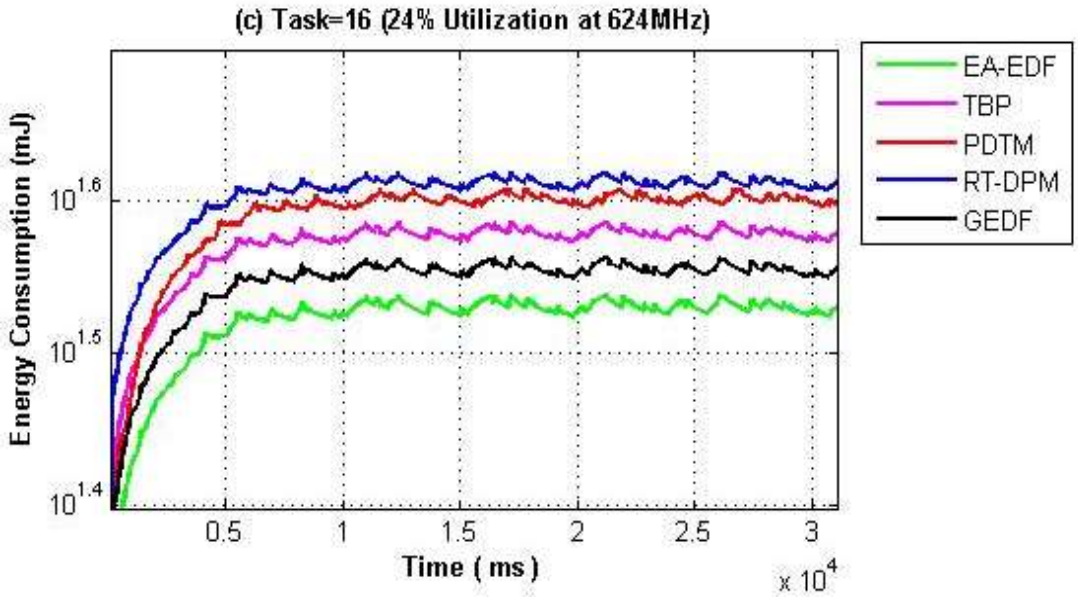


Figure 5. 4 *Comparison of energy consumption $u_i = 24$ % at 624 MHz*

Figure 5.4 represent the comparison of simulation conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= 24% with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, \dots . t_{21})$. A total of 4 cores are enough to schedule the task in an MPSoC. The X-axis is representing the time frame while Y-axis denotes the total energy consumption of our approach and other benchmarks in joules and supporting. Overall reduction in consumption of energy using EA-EDF occurs when the tasks are properly migrated to the least used core according to the variation in change in u_i utilization.

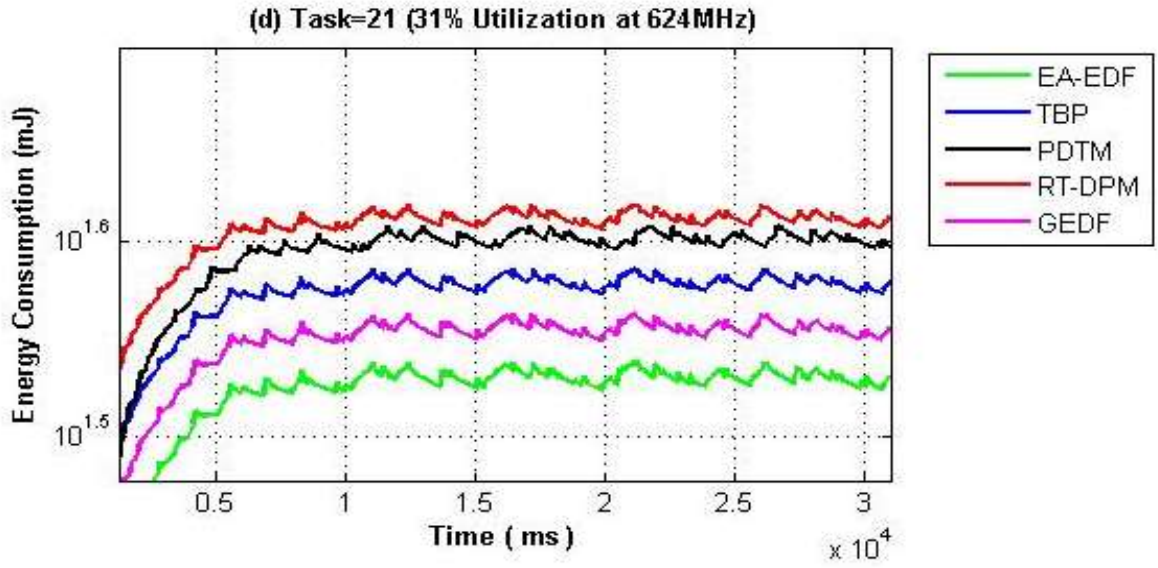


Figure 5. 5 *Comparison of energy consumption $u_i = 31$ % at 624 MHz*

Due to the availability of low-power processors in the core configurations for proper task mapping, substantial reduction in the energy consumption occurs as compared to other conventional techniques. Figure 5.5 represent the comparison of simulation conducted on simulation duration s_d=1000 ms using proposed EA.EDF at various workloads u_i= 31% with $m = 8$ cores $P = \{p_1, p_2, \dots, p_m\}$ having a periodic task set $\tau = (t_1, t_2, t_3, t_4, \dots . t_{25})$ total of 5 cores is enough to schedule the task in an MPSoC. The X-axis is representing the time frame while Y-axis denotes the total energy consumption of our approach and other benchmarks in joules and supporting

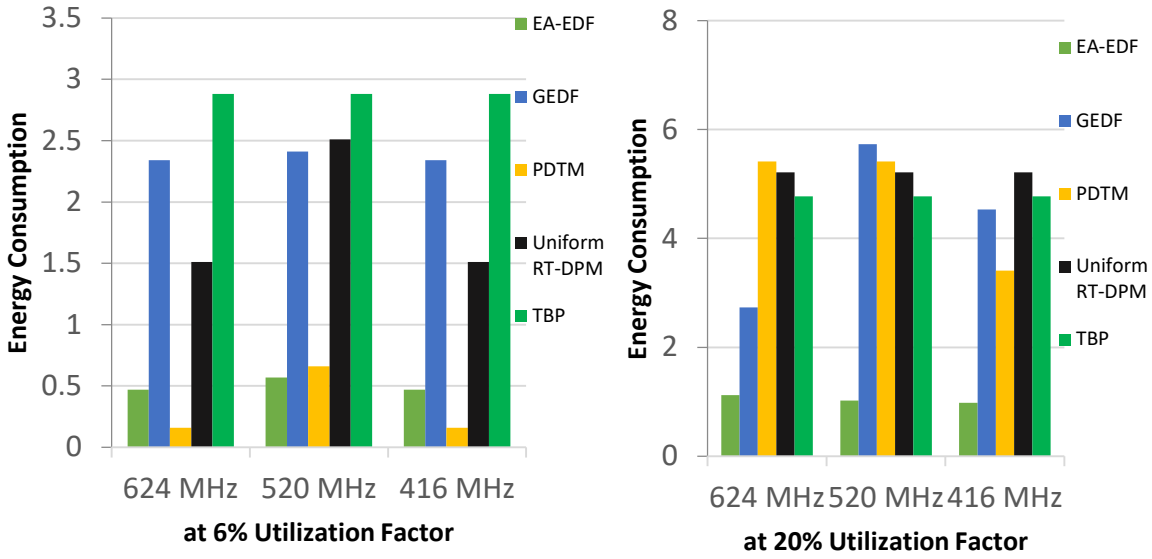


Figure 5. 6 *Comparison of energy consumption on* u_i *=6 & 20% at 624, 520, 416 MHz*

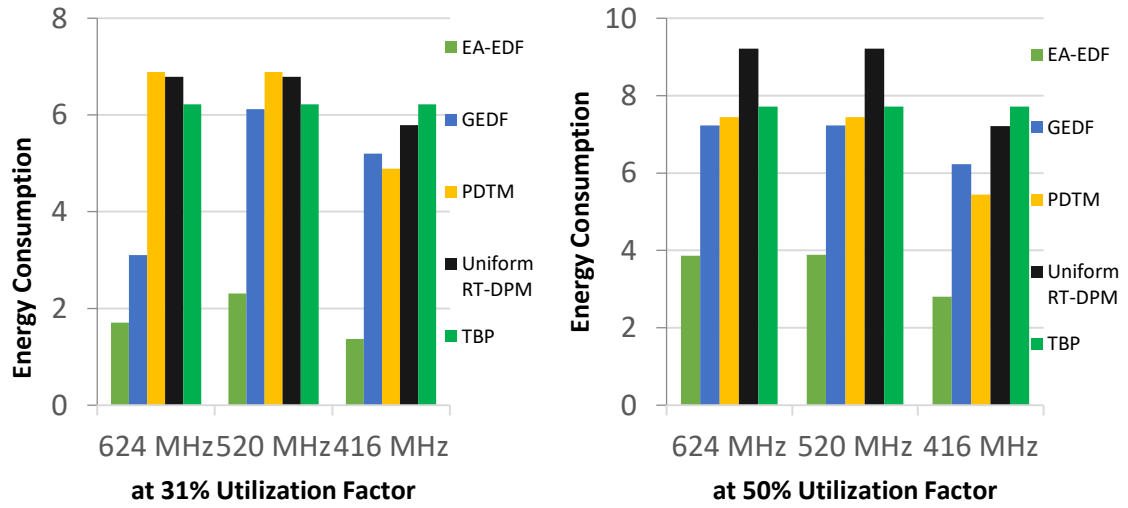


Figure 5. 7 *Comparison of energy consumption on* u_i *= 31% 50%,at* 624, 520, 416 MHz

Overall reduction in consumption of energy using EA-EDF occurs when the tasks are properly migrated to the least used core according to the variation in change in u_i utilization. Due to the availability of low-power processors in the core configurations for proper task mapping, substantial reduction in the energy consumption occurs as compared to other conventional techniques. Figure 5.6, and 5.7 represents the comparison of energy consumption using the proposed EA-EDF and the consumption of other conventional techniques G-EDF, PDTM U-RT-DPM, and TBP are evaluated at 624 MHz, 520 MHz and 416 MHz operating frequency. Each horizontal axis is representing the utilization factor that is increasing gradually with the arrival of the new task $t_i \in \tau$;

Each vertical axis represents the total consumption of energy. It is observed that with the increase in the number of tasks the overall utilization on chip increases due to efficient task migration and DPM-enabled policy. Due to the availability of low-power processors in the core configurations for proper task mapping significant reduction in energy consumption occurs. These graphs shows that with the decrease in operating frequency the energy reduction is significant on lower frequencies but some time low frequencies disturbs the performance that's why most of the INTEL PXA270 MPSoCs based embedded devices are operating at higher frequency standard at 624 MHz.

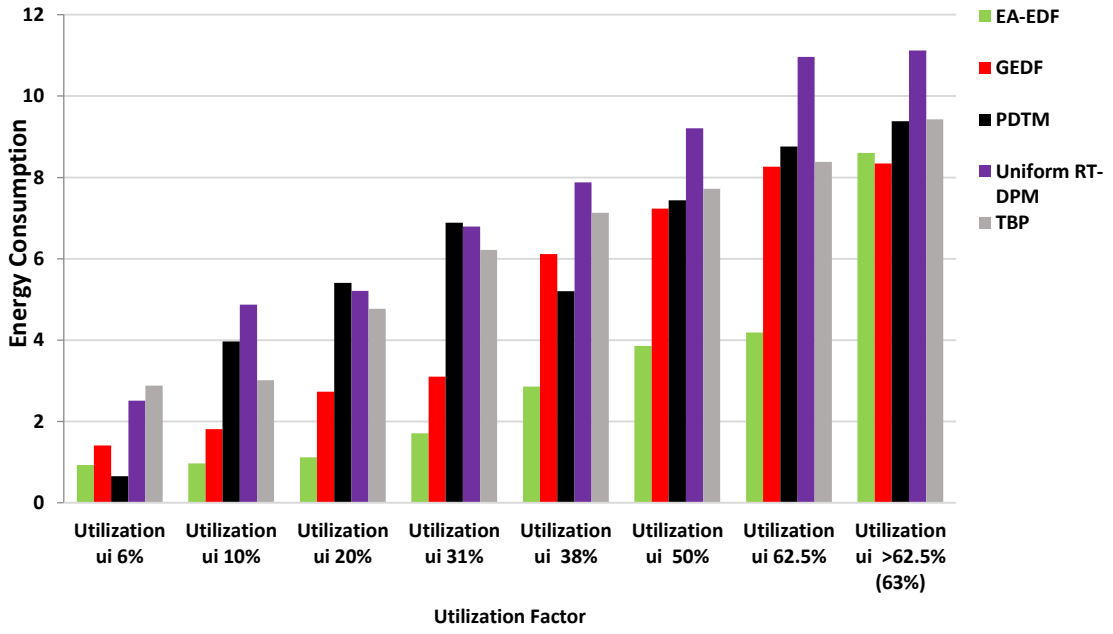


Figure 5. 8 *Proposed EA-EDF Energy Consumption Comparison with G-EDF, PDTM U-RT-DPM, TBP at 624MHz*

Figure 5.8 represents the energy efficiency of Intel PXA-270 MPSoC using the proposed EA-EDF and is further investigated with 4 other conventional techniques G-EDF, PDTM U-RT-DPM, and TBP at 624MHz.

Each horizontal X-axis is representing the utilization factor that is increasing gradually with the arrival of the new task $t_i \in \tau$; Each vertical Y-axis denotes the total energy consumption of our approach and other benchmarks in joules. It is observed that with the gradual increase in the accumulative number of tasks the overall utilization of chips increases due to efficient task migration and DPM-enabled policy.

Overall reduction in consumption of energy using EA-EDF occurs when the task is properly migrated to the least used core according to the variation in change in u_i utilization at u_i 31% of the application consume 2.86j and an increase in u_i 50 it consumes 3.86j. These energy values optimize 4.3J and 3.37J.

Due to the availability of low-power processors in the core configurations for proper task mapping, a significant reduction in energy consumption occurs. With the gradual increase in the accumulative number of tasks the overall utilization on chip increases due to efficient task migration and DPM-enabled policy. The impact of state switching and the low-power state and task mapping shows that the task migration feature guarantees the proper mapping of task while on the other hand (DPM) guarantees the lowest consumption of energy by managing the idle state to reduce the consumption.

Switching of CPU to low-power mode and task mapping shows that the task migration feature guarantees the proper task mapping while on the other hand (DPM) guarantees the lowest consumption of energy by managing the idle state to reduce the consumption.

Relatively EA-EDF reduces the consumption of energy using the Marvel INTEL PXA-270MPSoC platform containing multiple idle processors at lower utilization. Due to the availability of low-power processors in the core configurations for proper task mapping, substantial reduction in the energy consumption occurs as compared to other conventional techniques

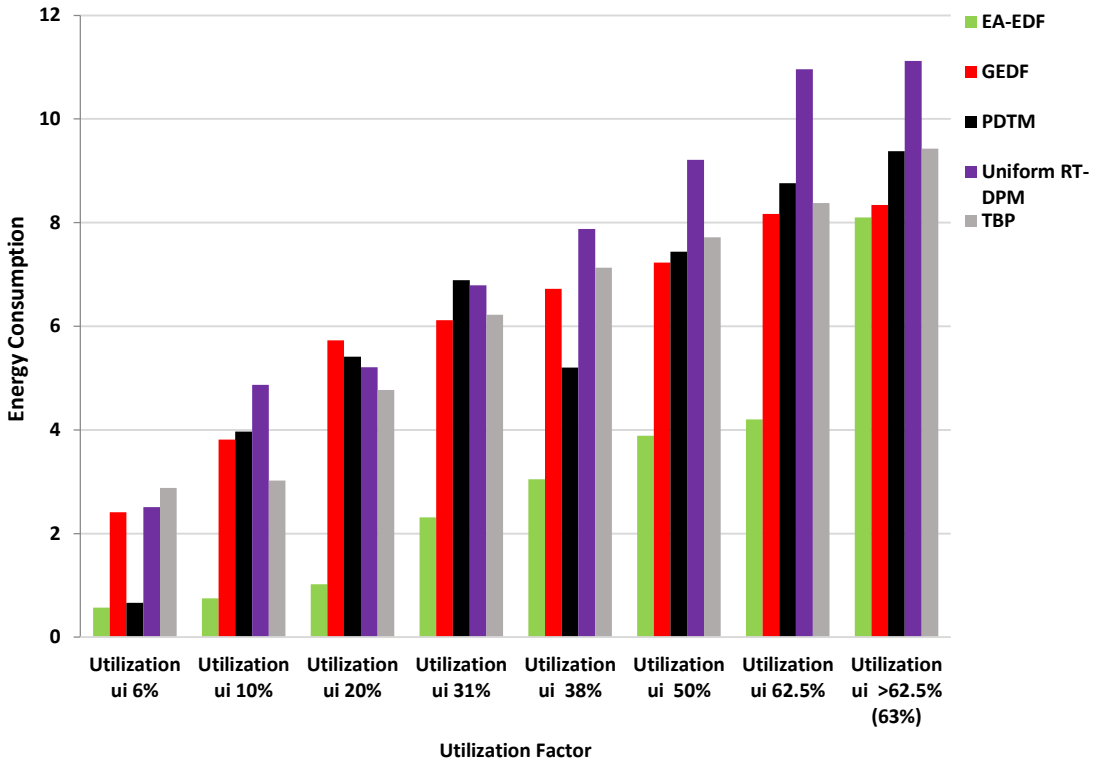


Figure 5. 9 *Proposed EA-EDF Energy Consumption Comparison with G-EDF, PDTM U-RT-DPM, TBP at 520MHz*

Figure 5.9 represents the energy efficiency of Intel PXA-270 MPSoC using the proposed EA-EDF and is further investigated with 4 other conventional techniques G-EDF, PDTM U-RT-DPM, and TBP at 520MHz. Each horizontal X-axis is representing the utilization factor that is increasing gradually with the arrival of the new task $t_i \in \tau$; Each vertical Y-axis denotes the total dynamic energy consumption of our approach and other benchmarks in joules. It is observed that with the increase in the number of tasks the overall utilization on chip increases due to efficient task migration and DPM-enabled policy. Switching of CPU to low-power mode and task mapping shows that the task migration feature guarantees the proper task mapping while on the other hand (DPM) guarantees the lowest consumption of energy by managing the idle state to reduce the consumption.

Overall reduction in consumption of energy using EA-EDF occurs when the task is properly migrated to the least used core according to the variation in change in u_i utilization at u_i 31% of the application consume 2.31j and an increase in u_i 50 it consumes 3.89j. These energy values optimize 4.58J and 4.7J. Due to the availability of low-power processors in the core configurations for proper task mapping significant

reduction in energy consumption occurs. Relatively EA-EDF reduces the consumption of energy using the Marvel INTEL PXA-270MPSoC platform containing multiple idle processors at lower utilization.

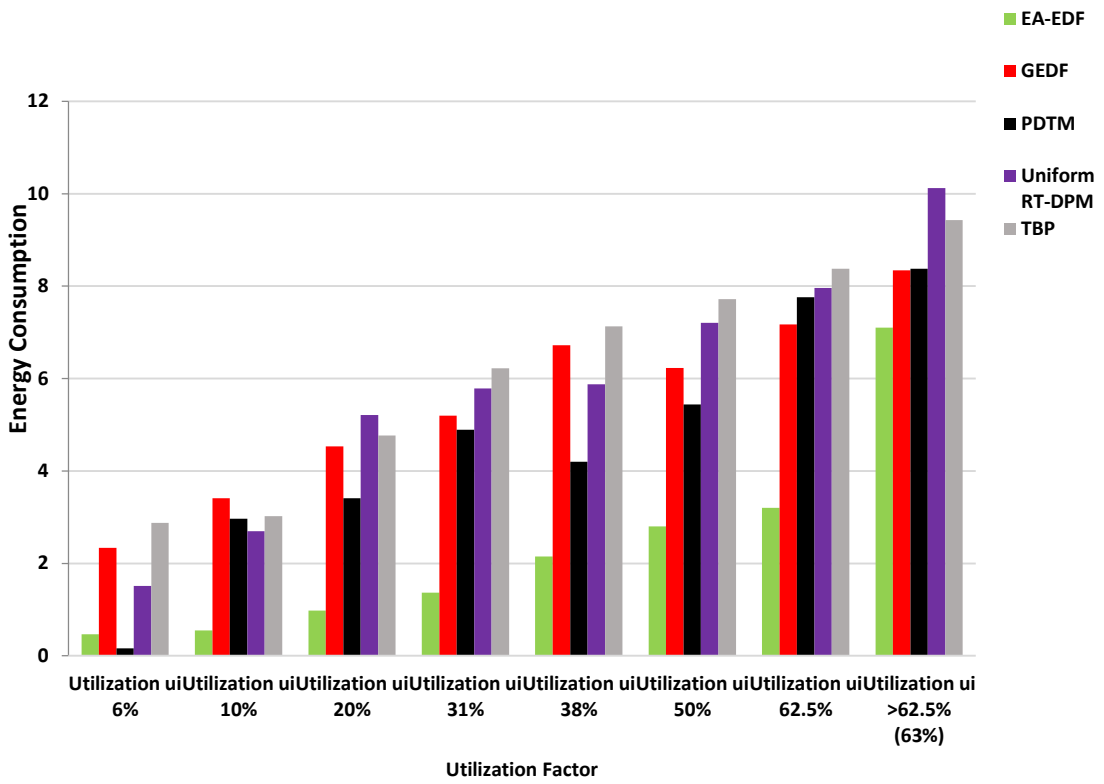


Figure 5. 10 *Proposed EA-EDF Energy Consumption Comparison with G-EDF, PDTM U-RT-DPM, TBP at 416MHz*

Figure 5.10 represents the energy efficiency of Intel PXA-270 MPSoC using the proposed EA-EDF and is further investigated with 4 other conventional techniques G-EDF, PDTM U-RT-DPM, and TBP at 416MHz. Each horizontal X-axis is representing the utilization factor that is increasing gradually with the arrival of the new task $t_i \in \tau$; Each vertical Y-axis denotes the total dynamic energy consumption of our approach and other benchmarks in joules. It is observed that with the increase in the number of tasks the overall utilization on chip increases due to efficient task migration and DPM-enabled policy.

The impact of the low-power state and task mapping shows that the task migration feature was supposed to guarantee the proper task mapping while on the other hand dynamic power management (DPM) guarantees the lowest consumption of energy by managing the idle state to reduce the consumption. Overall reduction in consumption of energy using EA-EDF occurs when the task is properly migrated to the least used core

according to the variation in change in u_i utilization at u_i 31% of the application consume 1.37j and an increase in u_i 50% it consumes 2.8j. These energy values optimize 4.23J and 5.3J. Due to the availability of low-power processors in the core configurations for proper task mapping significant reduction in energy consumption occurs. Relatively EA-EDF reduces the consumption of energy using the Marvel INTEL PXA-270MPSoC platform containing multiple idle processors at lower utilization.

5.4 Energy/Power Consumption & Task Scheduling using EA-EDF

The below section represents the consumption of power on various cores of MPSoCs using EA-EDF at $u_i = 6\%, 10\%, 20\%, 36\%, 55\%, 60\%, 62.5\%\ and\ u_i > 62.5\%$ with an increasing number of tasks τ at various configurations proposed in chapter 4; section 4.3.4.

5.4.1 Simulation Results at u_i $1-9\%$, $m = 1$

Using core configuration λ_1 s_d=1000 ms with quantum value ="10"we are evaluating the results on $u_i = 6\%$ using proposed EA-EDF.

$$\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (6\%),\ Core_{exe}(\alpha)=1,\ Core_{sleep}(\beta)=7 \quad (5.1)$$

$\forall t_i \in \tau\ t_1$ Period="6" activation Date="0" deadline="10" WCET="12" priority="1"

When the $u_i = 6\%$ then as per the proposed configuration λ_1 only one core is in running state and the rest of the seven cores in an octa-core processor are in sleep mode. Below results illustrates that the entire tasks τ are scheduled on core 1 without missing their deadlines on $u_i = 6\%$.

Figure 5.11 represents the exact simulation trace of a core configuration having one CPU core with a time interval 0-50 ms. As shown below in figure 5.13 all the task are properly scheduled on the single core and doesn't lead to violating the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using core configuration policy.

For validation of the simulation results using the EA-EDF scheduler the parameters for time in (ms) in the simulator are displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors. t_i activated simultaneously at time interval t= 0. $t_i \in \tau$ starts to schedule on m=1 CPU and the task $t_i = t_1$ with earliest d_i deadlines are ready to be executed first on $m = 1$ CPUs. As only one core is enough for the task load as its utilization is below 9% so there is no need to migrate the task or selection of other core configuration.

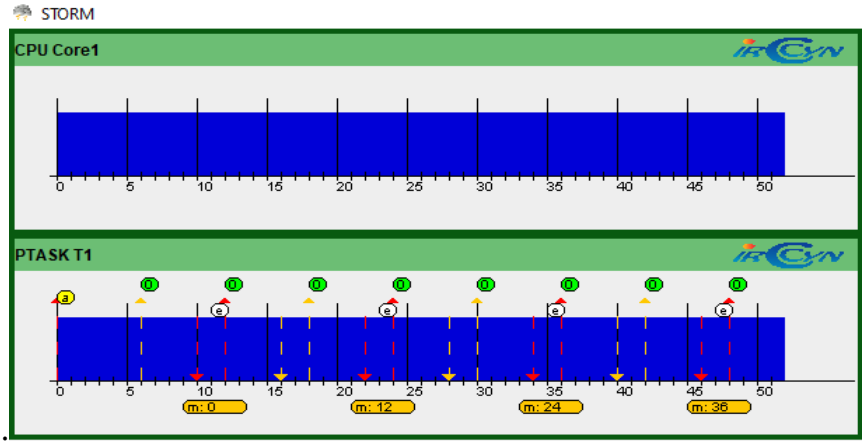
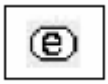
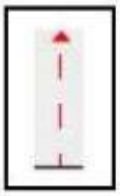


Figure 5. 11 *Gantt diagram scheduling of task on core 1 during simulation under* u_i*=6%*

 Represents the activation,  shows the completion and  shows the blocking and unblocking of a task ($t_i \in \tau$) and the number inside the circle shows the current status of block/unblock counter at u_i=6%.

 Represents the release and deadline (d_i)  of a periodic task $(t_i)\tau$

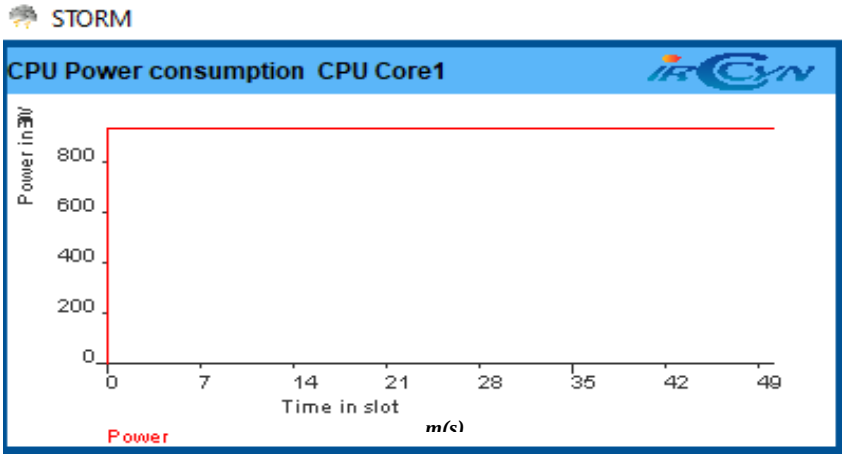


Figure 5. 12 *CPU Power Consumption on core 1 under* u_i*=6% at 624 MHz*

When ($u_i \leq \max threhold\ defined\ in\ \lambda_1$) then only one core configuration containing one CPU that meets the scheduling and u_i requirement of arrived task load ($t_i \in \tau$). Figure 5.12 represents the electrical power consumption for Intel PXA- 270 multiprocessor architecture with p=8 cores $P = \{p_1\}$ under u_i=6% at 624 MHz the

consumption of cores $\{p_1\}$ $P_{running}$ (α) =0.925 W in running. At u_i=6% CPU p_1 is completely occupied and not idle P_{idle} (Υ) at a single interval ensures no need for migration the $P_{running}$ $(\alpha) = 1$ executing the tasks and all the remaining cores are in $P_{sleep}(\beta)$ or idle state that causes the processor to reduce accumulative consumption of energy by keeping $P_{sleep}(\beta) = 7$ sleep mode using core configuration λ_1.

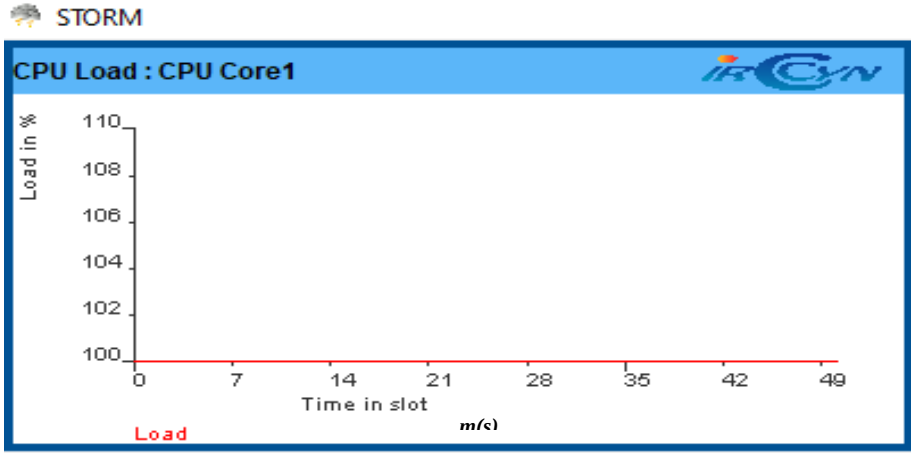


Figure 5. 13 *CPU load on core 1 under* u_i*=6% at 624 MHz*

5.4.2 Simulation Results at u_i 9.1 − 18%, $m = 2$

Using core configuration $\lambda_{2,}$ s_d=1000 ms we are evaluating the results on $u_i = 10\%$ using proposed EA-EDF.

$$\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (10\%)\,,\ Core_{exe}(s)=2,\ Core_{sleep}(s)=6 \qquad (5.2)$$

When the $u_i = 10\%$ then as per the proposed configuration $\lambda_{2,}$only two cores are in running state and the rest of the six cores in an octa-core processor are in sleep mode. Below results illustrates that the entire tasks τ are scheduled on (core 1) and (core 2) without missing their deadlines on $u_i = 10\%$.

Figure 5.14 represents the exact simulation trace of a core configuration having two CPU cores with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the multiprocessor and doesn't lead to violating the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy. Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$. For validation of the simulation results using the EA-EDF scheduler the parameters for time in (ms) in the simulator are displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue color represents the ready task of $t_1$and red color represents the ready task of t_2). At CPU core 1 task $t_i = t_2$ is activated

and scheduled on the CPU m_1 at time interval t= 0. The task $t_i = t_2$ is running and properly scheduled and is terminated at time instant t= 2 due to integration of dynamic power management DPM policy the CPU remains idle from time interval t= 2 ms to t= 4 ms. Another task $t_i \in \tau_{ready}$ queue $t_i = t_2$ is activated and scheduled on the CPU m_1 at time interval t= 4 ms till its absolute deadline on t= 14 ms.

At instant t=15 the ready task $t_i = t_2$ activated is still not terminated at time interval 16 ms and gets interrupted by another task $t_i = t_1$ as shown in CPU core 1 thus at that stage configuration with CPU core 2 is moved from idle to running and opting task migration policy to move the task $t_i = t_1$to m=2 core in the configuration that is not in use and simultaneously the task $t_i = t_1, t_2$ with earliest d_i deadlines are executed on $m = 2$ CPUs to avoid missing deadlines. Simulation results show that core configuration with more than one CPU supports to achieve the required quality of service and performance as these systems are based on energy consumption results in design exploration.

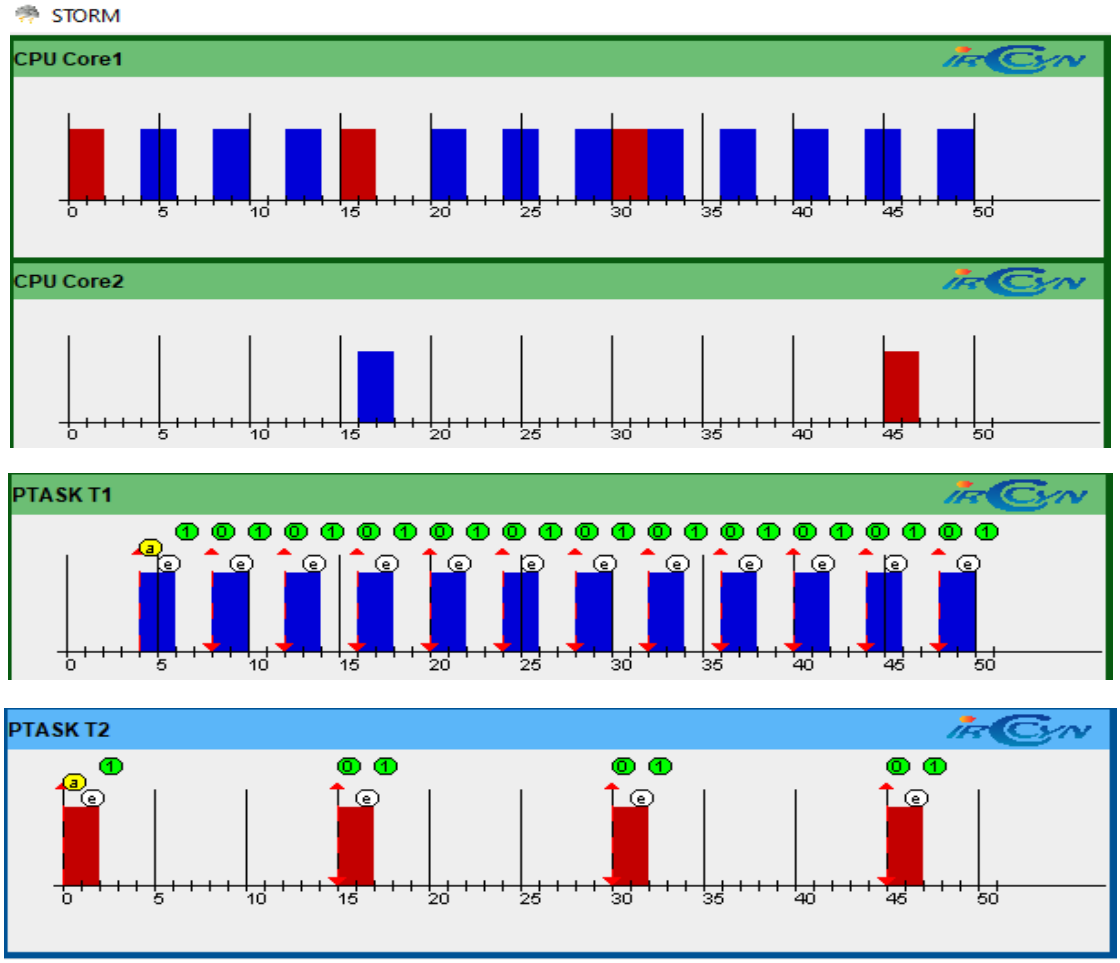


Figure 5. 14 *Gantt Chart of ready task set to be scheduled on core 1, core 2 core 1& core 2 during simulation under* u_i*=10% at 624 MHz*

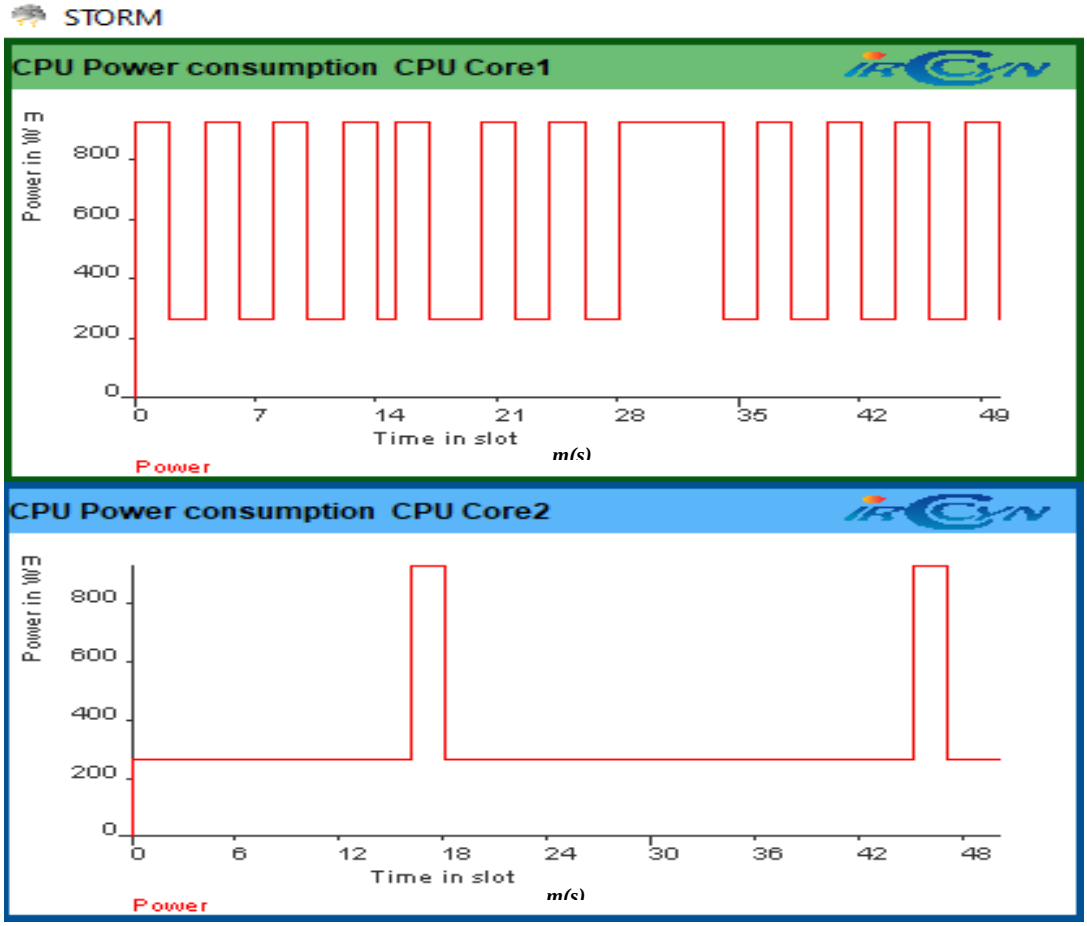


Figure 5. 15 *CPU Power Consumption on core* $\{\boldsymbol{p_1}, \boldsymbol{p_2}\}$ *at* u_i*=10% at 624 MHz*

When $(u_i \geq \max threhold\ defined\ in\ \lambda_1)$ then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable task load $(t_i \in \tau)$ and prior shifting the next core from $P_{sleep}(\beta)$ to P_{idle} (Υ) in the core configuration λ_2.

Figure 5.15 represents the electrical power consumption for Intel PXA- 270 multiprocessor architecture with p=8 cores $P = \{p_1, p_2\}$ CPU under u_i=10% at 624 MHz the consumption of cores $\{p_1, p_2\}$ $P_{running}$ (α) =0.925 W in running and consumption of P_{idle} (Υ) 0.26 W in idle mode.

At CPU p_2 it's visible that DPM shifts the core to a lower consumption state. The increase in the low power P_{idle} (Υ) idle duration at core 2 because EA-EDF uses task migration that schedules a task on the core in such an efficient way that less no of processing cores are in $P_{running}$ $(\alpha) = 2$ executing the task and more cores remain in $P_{sleep}(\beta)$ or idle state that causes the processor to decrease the consumption of energy by keeping $P_{sleep}(\beta) = 6$ sleep mode using core configuration λ_2.

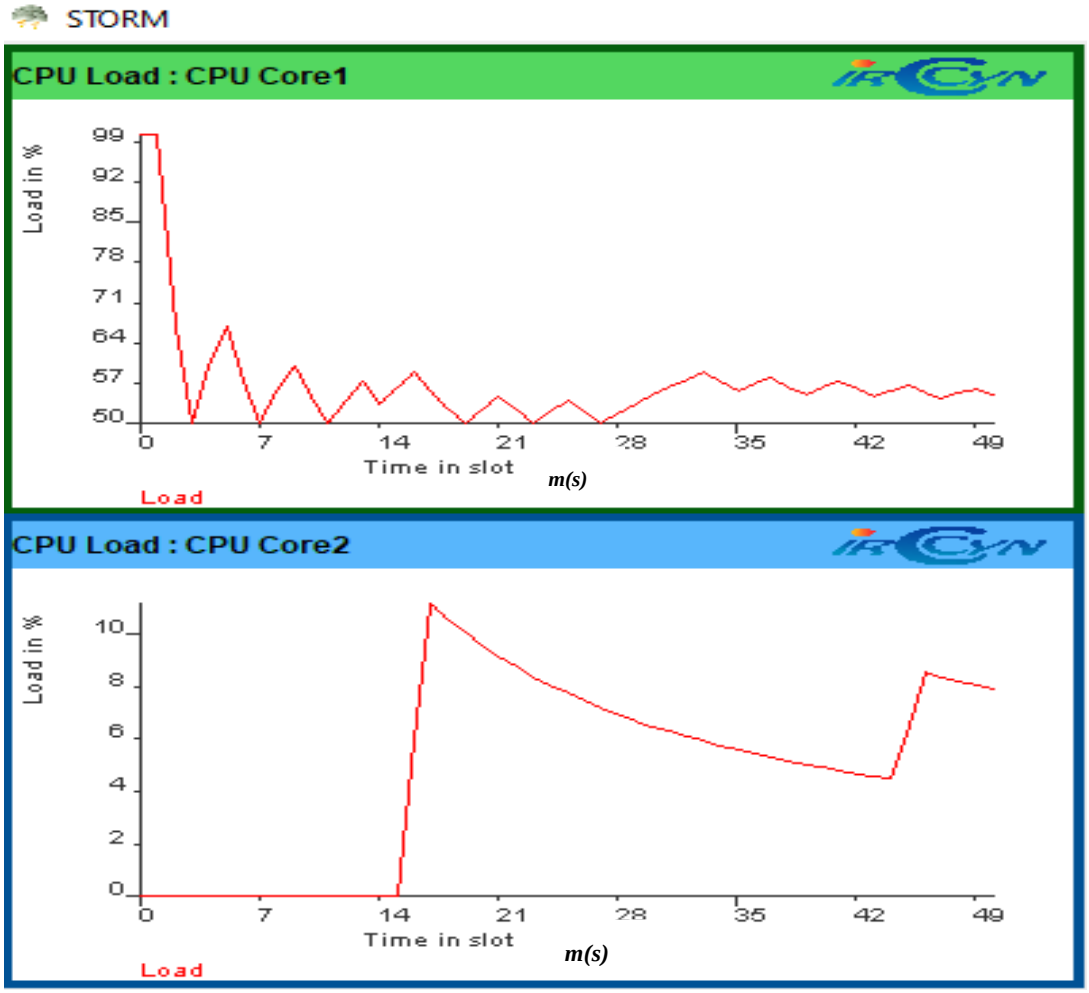


Figure 5. 16 *CPU load on core 1 under* u_i*=10% at 624 MHz*

5.4.3 Simulation Results at u_i 18. 1 − 27%, $m = 3$

Using core configuration λ_3, s_d=1000 ms we are evaluating the results on $\boldsymbol{u_i} = 20\%$ using proposed EA-EDF.

$$\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (20\%)\,,\ Core_{exe}(s)=3,\ Core_{sleep}(s)=5 \tag{5.3}$$

When the $\boldsymbol{u_i} = 20\%$ then as per the proposed configuration λ_3 only three cores are in running state and the rest of the five cores remain in sleep mode. Below results illustrates that the entire tasks τ are scheduled on (core 1), (core 2) and (core 2) without missing their deadlines on $u_i = 20\%$.

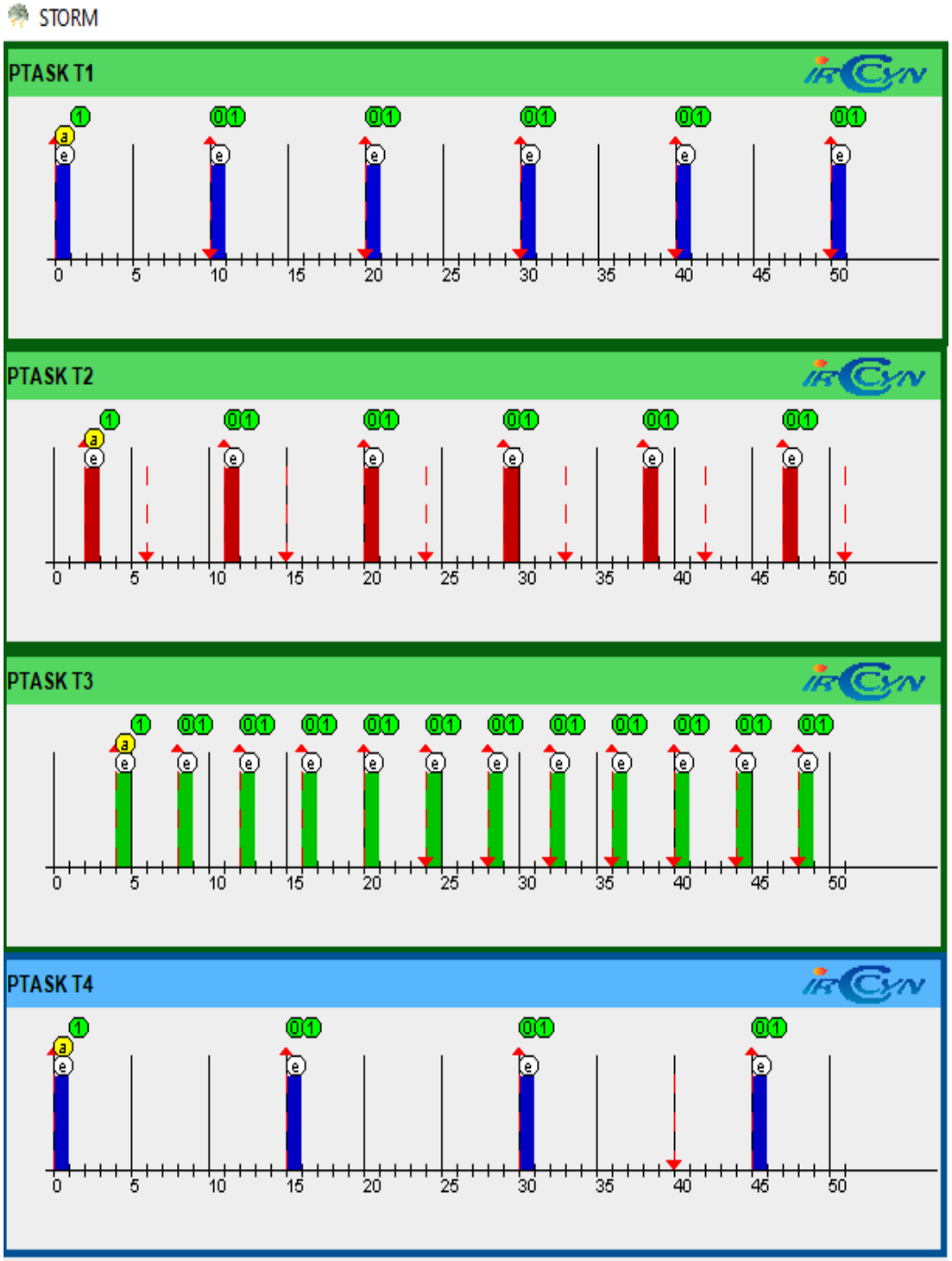


Figure 5. 17 *Gantt Chart of ready task set to be scheduled on core 1, core 2 & core 3 under* u_i*=20% at 624 MHz*

Figures 5.17 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using the gantt chart that requires a CPU to schedule. Figure 5.18 The exact simulation trace of a core configuration having three CPU cores with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the

multiprocessor and doesn't lead to violating the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy.

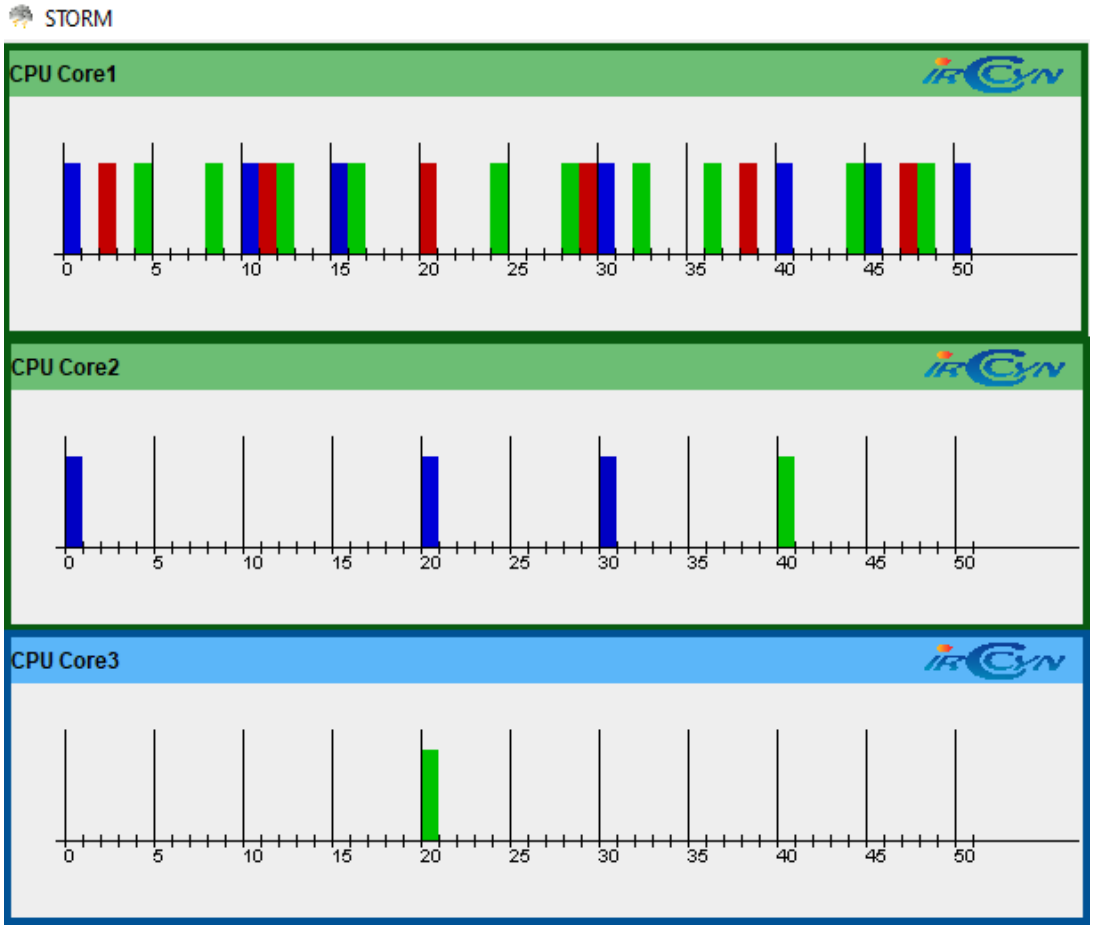


Figure 5. 18 *Scheduling of tasks using the EA-EDF technique for 3-processor configuration under* u_i*=20%*

Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$. For validation of the simulation results using the EA-EDF scheduler the parameters for time in (ms) in the simulator are displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue color represents the ready task of $t_1$and the red color represents the ready task of t_2 while the green color represents the ready task of t_3).At CPU core 1 task $t_i = t_1$ is activated and scheduled on the CPU m_1 at time interval t= 0. The task $t_i = t_1$ is running and properly scheduled and is terminated at time instant t= 1 due to integration of dynamic power management DPM policy the CPU remains idle from time interval t= 1 ms to t= 2 ms. Another task $t_i \in \tau_{ready}$ queue $t_i = t_2$ is activated and scheduled on CPU m_1 at time interval t= 2 ms till its absolute deadline on t= 3ms due to dynamic power management DPM policy the CPU remains idle from time interval t= 3 ms to t= 4 ms. At time instant t=4 the ready

task $t_i = t_3$ activated and terminated at time instant 5 ms and the scheduling continues but at interval t= 20 ms $t_i = t_2$ is running on CPU core 1 and another task $t_i = t_4$ and $t_i = t_3$ arrived at the same time frame as thus at that stage configuration with CPU core 2 and core 3 is moved from idle to running and opting task migration policy to move the task $t_i = t_4$ to m=2 core and $t_i = t_3$ to m=3 core in the configuration that is not in use and simultaneously the task $t_i = t_1, t_2, t_3, t_4$ with earliest d_i deadlines are executed on $m = 1, 2$ and 3 CPUs to avoid missing deadlines.

Simulation results show that core configuration with more than two CPUs satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration.

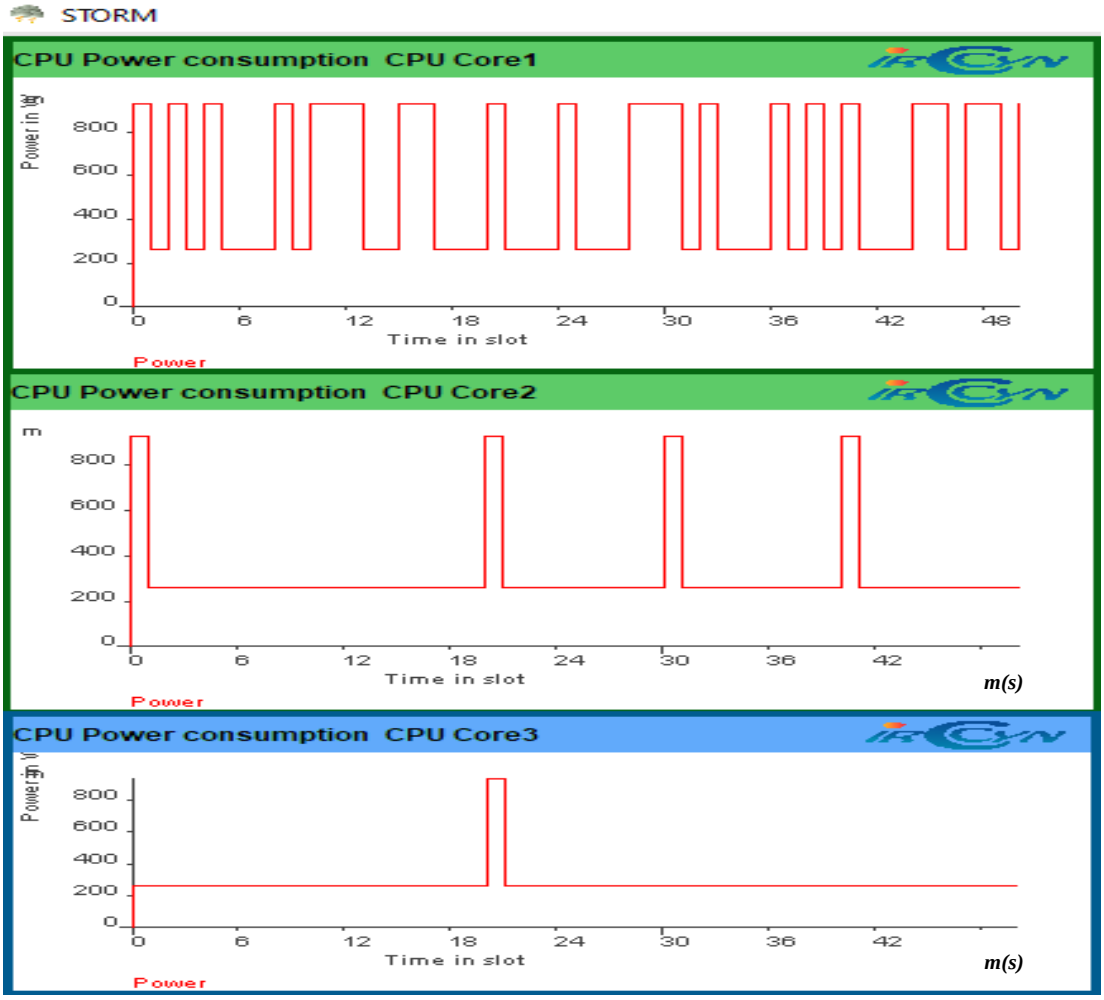


Figure 5. 19 *CPU Power Consumption on core p_1, p_2, p_3 under u_i=20% at 624 MHz*

When $(u_i \geq \max threhold\ defined\ in\ \lambda_2)$ then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable

task load ($t_i \in \tau$) and prior shifting the next core from $P_{sleep}(\beta)$ to P_{idle} (Υ) in the core configuration λ_3. Figure 5.19 represents the electrical power consumption for Intel PXA-270 multiprocessor architecture with p=8 cores $P = \{p_1, p_2, p_3\}$ under u_i=20% at 624 MHz the consumption of $P_{running}(\alpha)$ =0.925 W in running and consumption of P_{idle} (Υ) 0.26 W in idle mode. At CPU core $\{p_2, p_3\}$ it's visible that DPM shifts the core to a lower energy, power consumption state (Υ). The increase in the low power P_{idle} (Υ) idle duration at $\{p_2, p_3\}$ is because EA-EDF uses task migration that schedules a task on the core in such an efficient way that less no of processing cores are in $P_{running}(\alpha) = 3$ executing the task and more cores remain in $P_{sleep}(\beta)$ or idle state that causes the CPU to decrease the total energy consumption by keeping $P_{sleep}(\beta) = 5$ sleep mode using core configuration λ_3.

Figure 5. 20 *CPU load on core 1, 2 & 3 under u_i=20% at 624 MHz*

Figure 5.20 represents the CPU load shows the scheduling tasks on various cores. The higher peak of the curve shows that multiple tasks $t_i \in \tau$ requires the CPU time to

schedule without missing their $\boldsymbol{d_i}$ increases the utilization $\boldsymbol{u_i}$. The fall of the curve shows that DPM enhances the idle duration to maintain a low power mode that results in lower energy consumption using EA-EDF.

5.4.4 Simulation Results at $\boldsymbol{u_i}$ $\boldsymbol{27.1-36\%}$, $\boldsymbol{m=4}$

Using core configuration λ_4, s_d=1000 ms we are evaluating the results on $\boldsymbol{u_i} = 31\%$ using proposed EA-EDF.

$$\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (31\%)\,,\ Core_{exe}(s)=4,\ Core_{sleep}(s)=4 \tag{5.4}$$

Figure 5. 21 *Scheduling of tasks using the EA-EDF technique for 4-processor configuration*

When the $u_i = 31\%$ then as per the proposed configuration 4 only four cores are in running state and the rest of the four cores in sleep mode. Below results illustrates that the entire tasks τ are scheduled on (CPU core 1, CPU core 2, CPU core 3 and CPU core 4) without missing their deadlines on $u_i = 31\%$.

Figures 5.21 The exact simulation trace of core configuration having four CPU core with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the multiprocessor and doesn't lead to violate the deadline for execution of the ready tasks from task set $\forall t_i \in \tau$ using proposed core configuration policy.

Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$. For validation of the simulation results using EA-EDF scheduler the parameters for time in (ms) in the simulator is displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue colour represents the ready task of $t_1$and red colour represents the ready task of t_2 while green colour represents the ready task of t_3).

At CPU core 1 task $t_i = t_1$ is activated and scheduled on CPU m_1 at time interval t= 0. The task $t_i = t_1$ is running and properly scheduled and is terminated at time instant t= 1 due to integration of dynamic power management DPM policy the CPU remains idle from time interval t= 1 ms to t= 2 ms. Another task $t_i \in \tau_{ready}$ queue $t_i = t_2$ is activated and scheduled on CPU m_1 at time interval t= 2 ms till its absolute deadline on t= 3ms due to dynamic power management DPM policy the CPU remains idle from time interval t= 3 ms to t= 4 ms.

At time instant t=4 the ready task $t_i = t_3$ activated and terminated at time instant 5 ms and the scheduling continues but at interval t= 20 ms $t_i = t_2$ is running on CPU core 1 and another task $t_i = t_5$ $t_i = t_4$ and $t_i = t_3$ arrived at the same time frame as thus at that stage configuration with CPU core 2, core 3, core 4 is moved from idle to running and opting task migration policy to move the task $t_i = t_4$ to m=2 core, $t_i = t_5$ to m=4 core and $t_i = t_3$ to m=3 in the configuration that is not in use and simultaneously the task $t_i = t_1, t_2, t_3, t_4$ with earliest d_i deadlines are executed on $m = 1, 2$ and 3 CPUs to avoid missing deadlines.

Simulation results shows that core configuration with more than three CPU satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration.

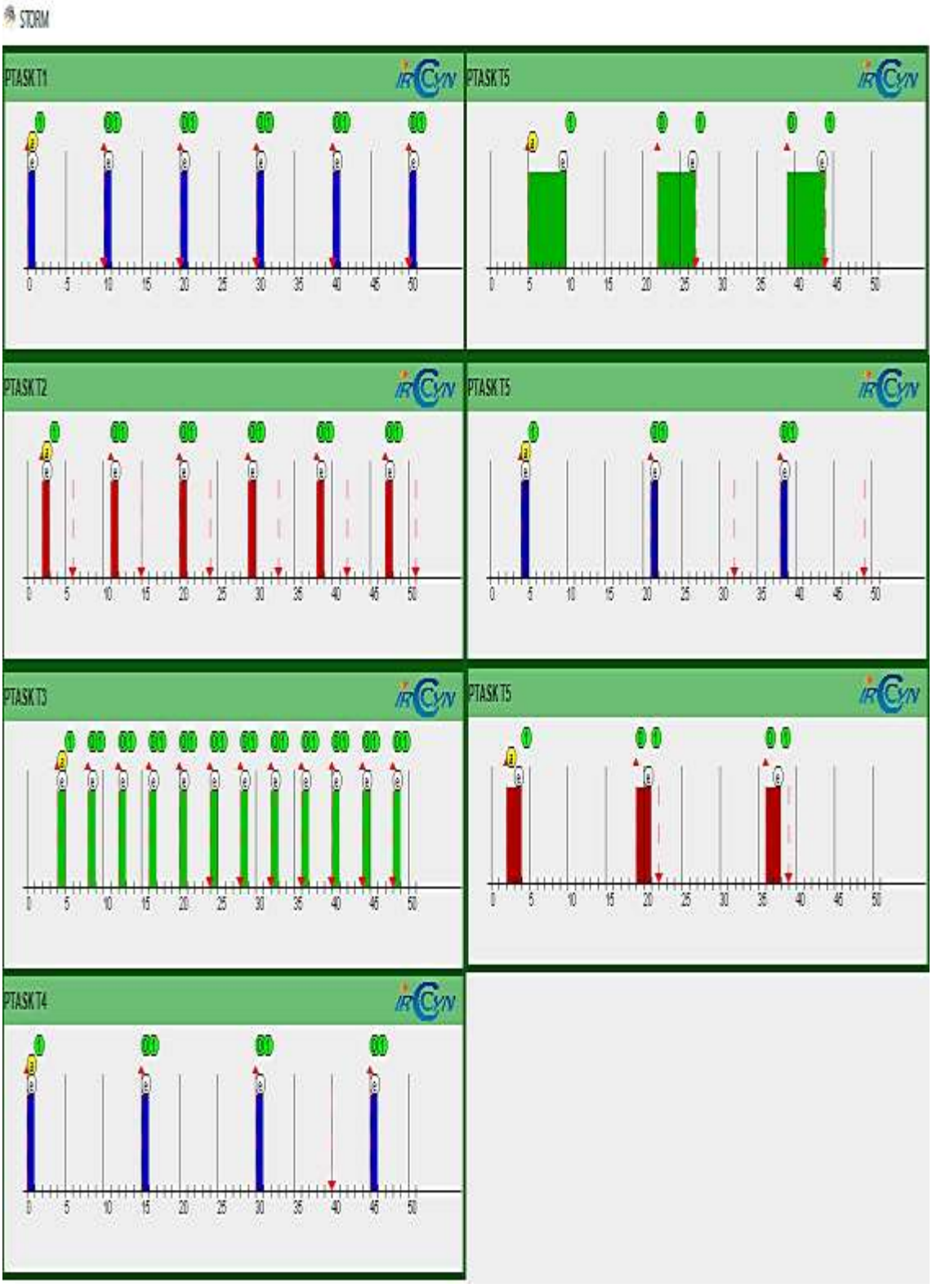


Figure 5. 22 *Gantt Chart of ready task set to be scheduled on core 1, 2, 3 and 4 under u_i=28% at 624 MHz*

Figures 5.22 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using Gantt chart that requires a CPU to schedule.

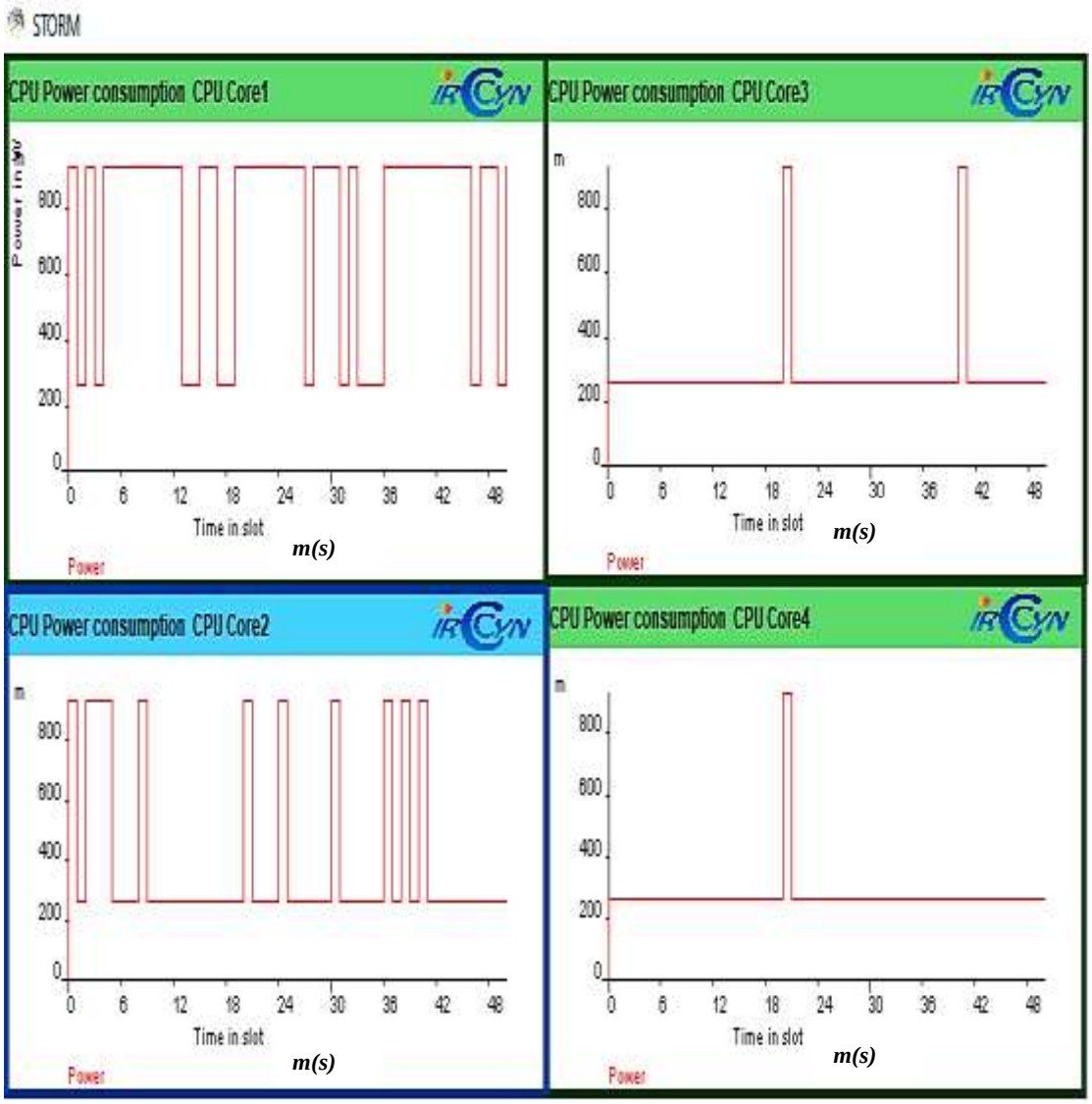


Figure 5. 23 *CPU Power Consumption on p_1, p_2, p_3, p_4 under u_i=31% at 624 MHz*

When $(u_i \geq \max threhold\ defined\ in\ \lambda_3)$ then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable task load $(t_i \in \tau)$ and prior shifting the next core from $P_{sleep}(\beta)$ to P_{idle} (Υ) in the core configuration λ_4.

Figure 5.23 represents the electrical power consumption of Intel PXA- 270 multiprocessor architecture with m cores $P = \{p_1, p_2, p_3, p_4\}$ under u_i=31% at 624 MHz the consumption of cores $\{p_1, p_2, p_3, p_4\}$ $P_{running}(\alpha)$ =0.925 W in running and consumption of P_{idle} (Υ) 0.26 W in idle mode. At CPU cores $\{p_1, p_2, p_3, p_4\}$ it's visible that DPM shifts all the core to a lower power consumption state (Υ) by increasing the low power P_{idle} (Υ) idle duration at core $\{p_2, p_3\}$ is because EA-EDF uses task migration that schedules a task on the core in such an efficient way that less no of

processing cores are in $P_{running}\ (\alpha) = 4$ executing the task and more cores remain in $P_{sleep}(\beta)$ or idle state that causes the CPU cores to educe accumulative consumption of energy by keeping $P_{sleep}(\beta) = 4$ sleep mode using core configuration λ_4.

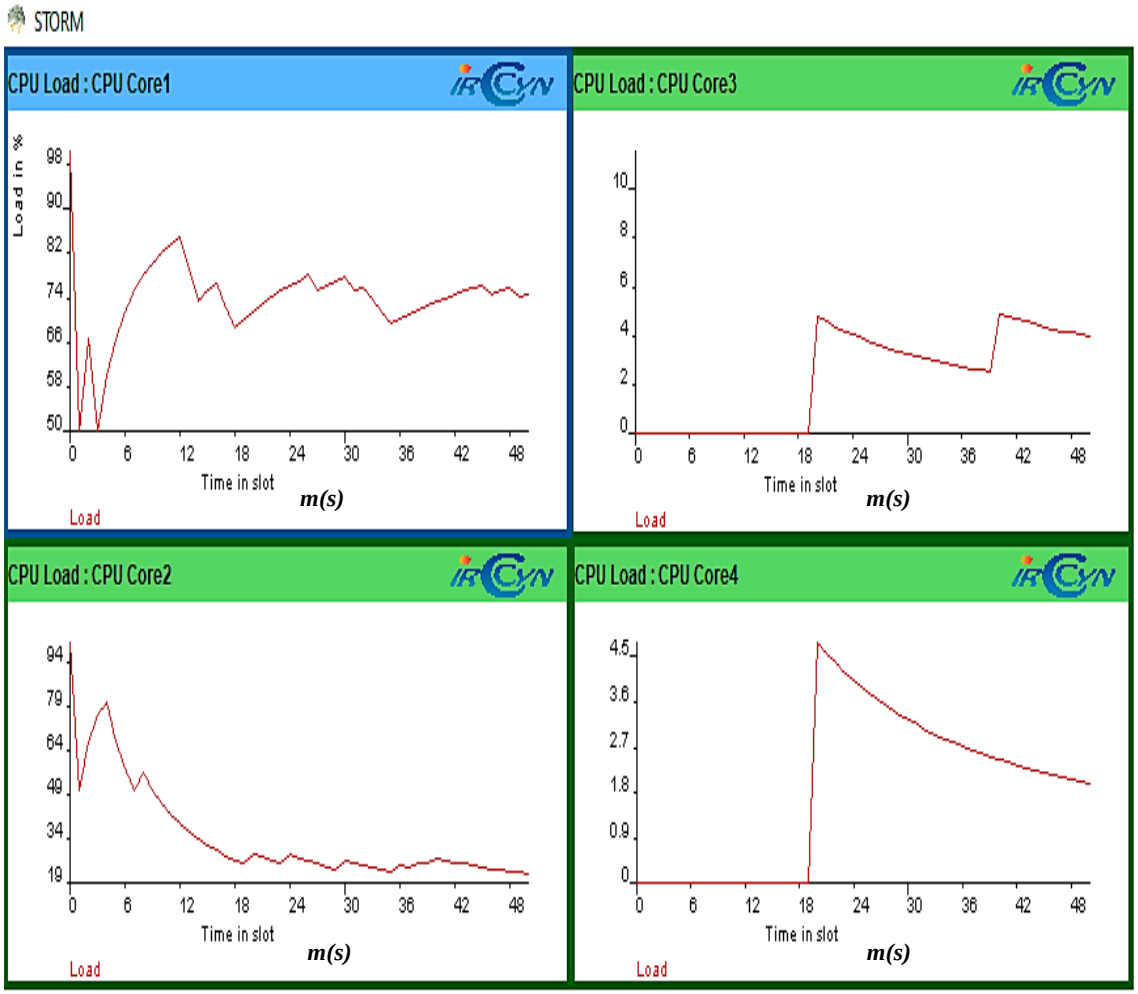


Figure 5. 24 *CPU load on core 1, core 2, core 3 & core 4 under* u_i*=31% at 624 MHz*

Figure 5.24 represents the CPU load shows the scheduling tasks on various cores. The higher peak of the curve shows that multiple tasks $\boldsymbol{t_i} \in \boldsymbol{\tau}$ requires the CPU time to schedule without missing their $\boldsymbol{d_i}$ increases the utilization $\boldsymbol{u_i}$. The fall of the curve shows that DPM enhances the idle duration to maintain a low power mode that results in lower energy consumption using EA-EDF.

5.4.5 Simulation Results at $\boldsymbol{u_i}$ $\boldsymbol{36.1 - 45\%, m = 5}$

Using core configuration λ_5, s_d=1000 ms we are evaluating the results on $\boldsymbol{u_i} = 38\%$ using proposed EA-EDF.

$$\forall \tau \to u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (38\%)\,,\ Core_{exe}(s)=5,\ Core_{sleep}(s)=3 \qquad (5.5)$$

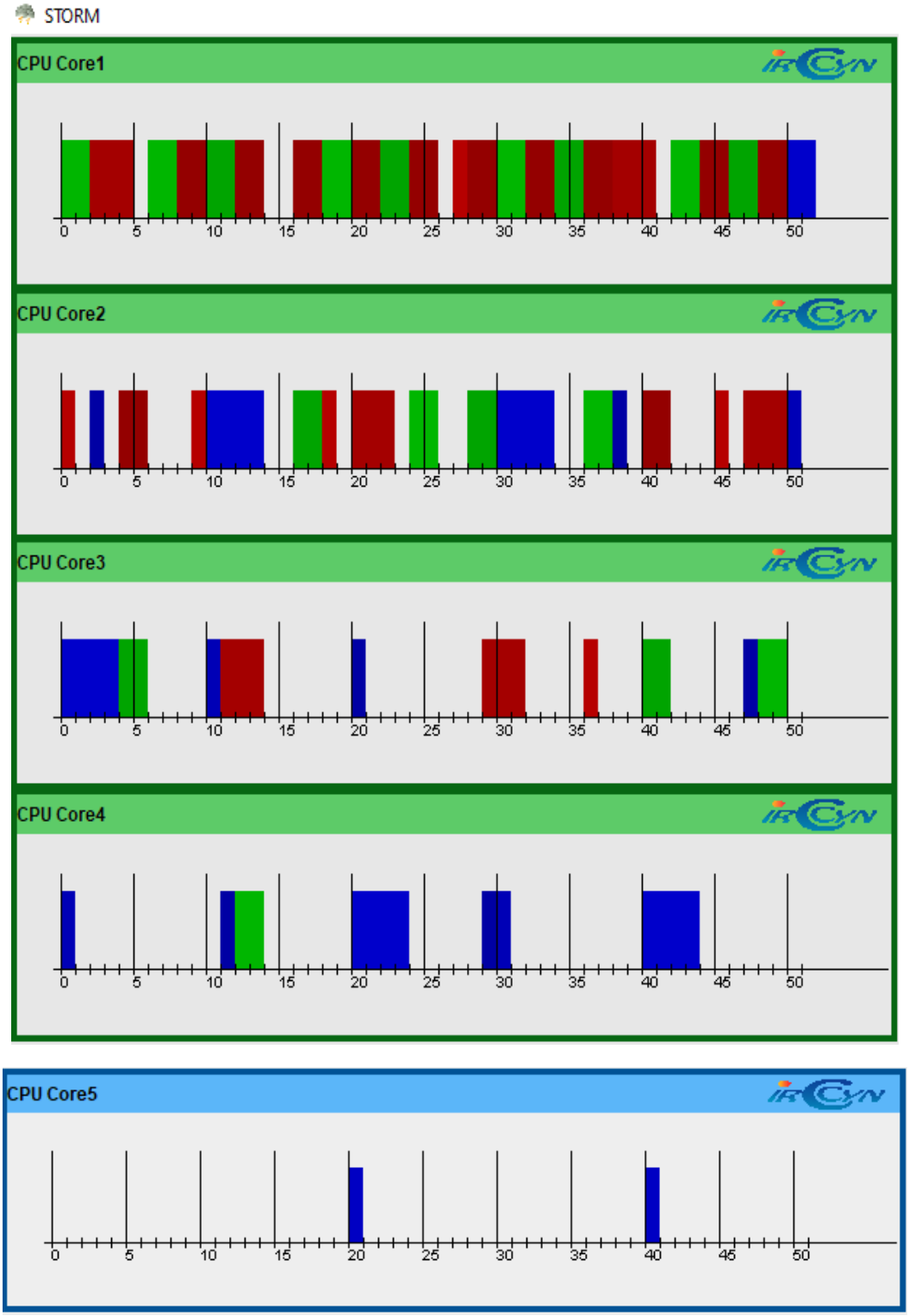


Figure 5. 25 *Scheduling of tasks using the EA-EDF technique for 5-processor configuration under* u_i*=38% at 624 MHz*

When the $u_i = 38\%$ then as per the proposed configuration 5 only five cores are in running state and the rest of the three cores in an octa-core processor are in sleep mode. Below results illustrates that the entire tasks τ are scheduled on processor (core m=1,

core 2, core 3, core 4 and CPU core 5) without missing their deadlines on $u_i = 38\%$. Figures 5.25 represent the exact simulation trace of core configuration having five CPU core with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the multiprocessor and doesn't lead to violate the deadline for execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy. Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$.

For validation of the simulation results using EA-EDF scheduler the parameters for time in (ms) in the simulator is displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue color represents the ready task of $t_1$and the red color represents the ready task of t_2 while the green color represents the ready task of t_3 and so on). At CPU core 1 task $t_i = t_3$ is activated and scheduled on the CPU m_1 at time interval t= 0. The task $t_i = t_3$ is running and properly scheduled on CPU Core 2. Another task $t_i \in \tau_{ready}$ queue $t_i = t_1$ is activated and scheduled on the CPU m_1 at time interval t= 0 ms till its absolute deadline on t= 4ms on CPU core 3 because of task migration and core selection policy all the task arriving at interval t= 0 is scheduled properly without missing their deadlines and once the scheduled task is executed on the core will remain in idle mode due to dynamic power management DPM policy the CPU core 1 remains idle from time interval t= 4 ms to t= 4 ms. CPU core 2 is moved to idle at time intervals t= 2 ms to t= 3 ms. CPU core 3 is moved to idle at time intervals t=6 ms to t= 10 ms. CPU core 4 is moved to idle at time interval t=2 ms to t= 11 ms. CPU core 5 is moved to idle at time interval t=0 ms to t= 20 ms, from t=21 ms to t= 40 ms and from t=41 ms to t= 50 ms. Task $t_i = t_5$ $t_i = t_4$, $t_i = t_3$, $t_i = t_2$ and $t_i = t_1$ arrived at the same time frame time interval t=20 ms as thus at that stage configuration with CPU core m=2, core m=3, core m=4 and core m=5 is moved from idle to running and opting task migration policy to move the task $t_i = t_4$ to m=2 core, $t_i = t_5$ to m=4 core, $t_i = t_2$ to m=3 and $t_i = t_3$ to m=3 in the configuration that is not in use and simultaneously the task $t_i = t_1, t_2, t_3, t_4, t_5$ with earliest d_i deadlines are executed on $m = 1, 2, 3, 4$ and 5 CPUs to avoid missing deadlines. Simulation results show that core configuration with more than four CPUs satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration. Figures 5.26 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using gantt chart that requires a CPU to schedule.

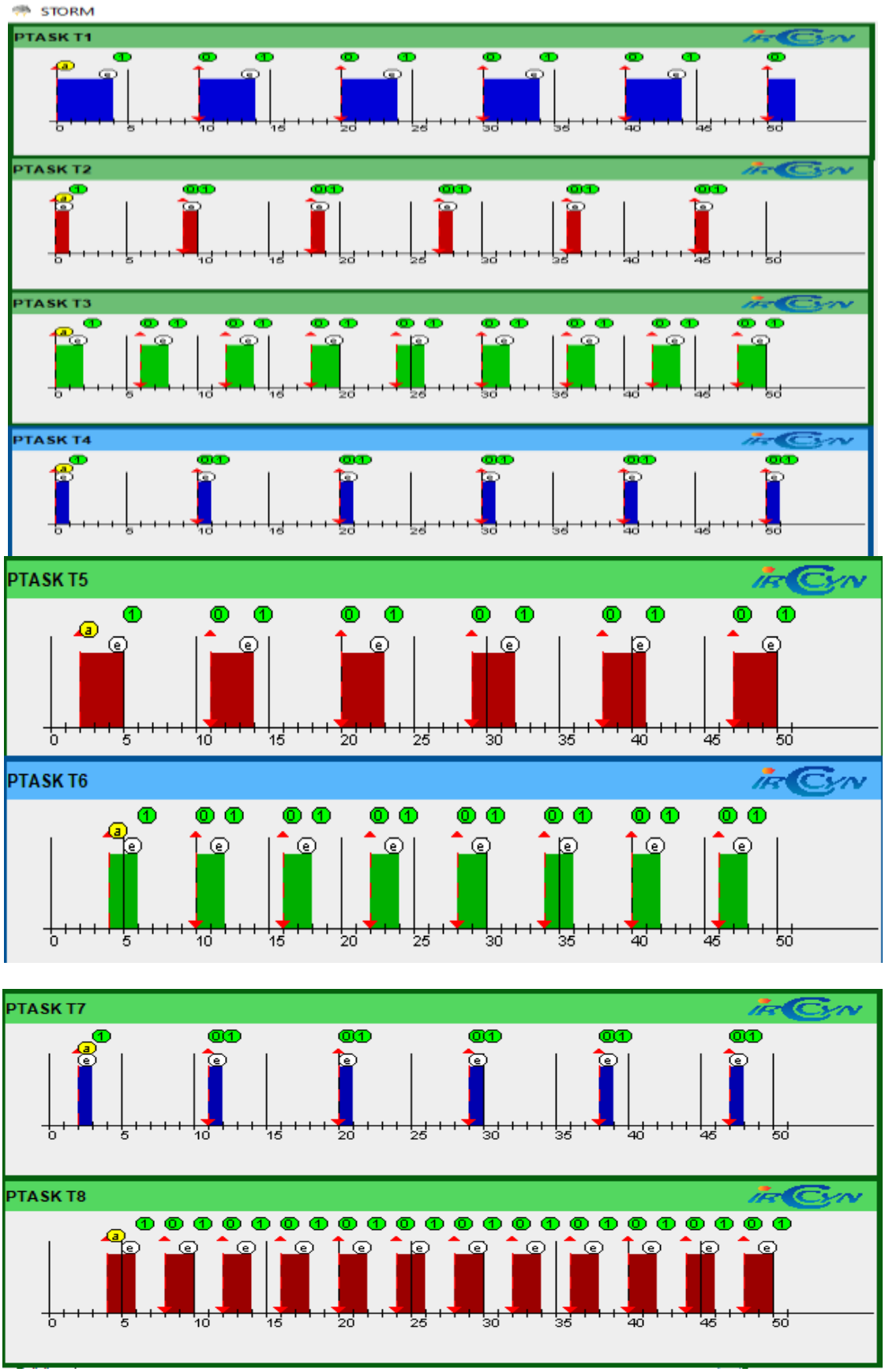


Figure 5. 26 *Gantt charts for scheduling of tasks on core p_1, p_2, p_3, p_4, p_5during simulation under u_i=38%*

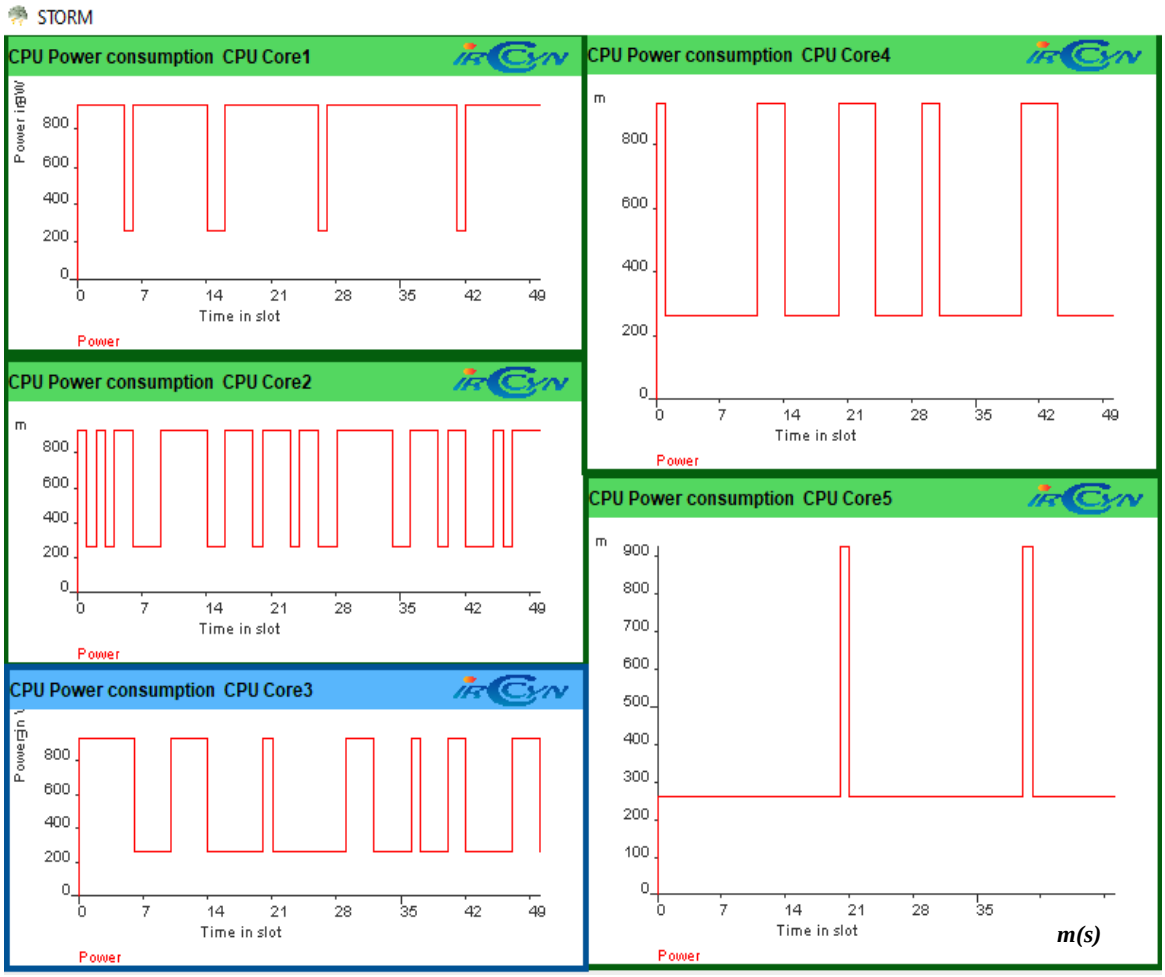


Figure 5. 27 *CPU Power Consumption on core* $p_1, p_2, .., p_5$ *under* u_i *=38% at 624MHz*

Figure 5.27 represents the electrical power consumption for PXA- 270 multiprocessor architecture with m cores $P = \{p_1, p_2, p_3, p_4, p_5\}$ under u_i=31% at 624 MHz. When ($u_i \geq \max threhold\ defined\ in\ \lambda_4$) then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable task load ($t_i \in \tau$) and prior shifting the next core from $P_{sleep}(\beta)$ to $P_{idle}(\Upsilon)$ in the core configuration λ_5. The consumption of cores {1,2,3,4} $P_{running}(\alpha)$ =0.925 W in running and consumption of $P_{idle}(\Upsilon)$ 0.26 W in idle mode. At CPU cores $\{p_1, p_2, p_3, p_4, p_5\}$ it's visible that DPM shifts all the core to a lower power consumption state (Υ) by increasing the low power $P_{idle}(\Upsilon)$ idle duration specifically at core $\{p_1, p_3, p_4, p_5\}$is because EA-EDF uses task migration that schedules a task on the cores efficiently and guarantees to meet the deadlines $P_{running}(\alpha) = 5$ executing the task and fewer cores remain in $P_{sleep}(\beta)$ or idle state that causes the processor to reduce accumulative consumption of energy by keeping $P_{sleep}(\beta) = 3$ sleep mode using core configuration λ_5. Figure 5.28 represents the CPU load shows the scheduling tasks on

various cores. The higher peak of the curve shows that multiple tasks $t_i \in \tau$ requires the CPU time to schedule without missing their d_i increases the utilization u_i. The fall of the curve shows that DPM enhances the idle duration to maintain a low power mode that results in lower energy consumption using EA-EDF.

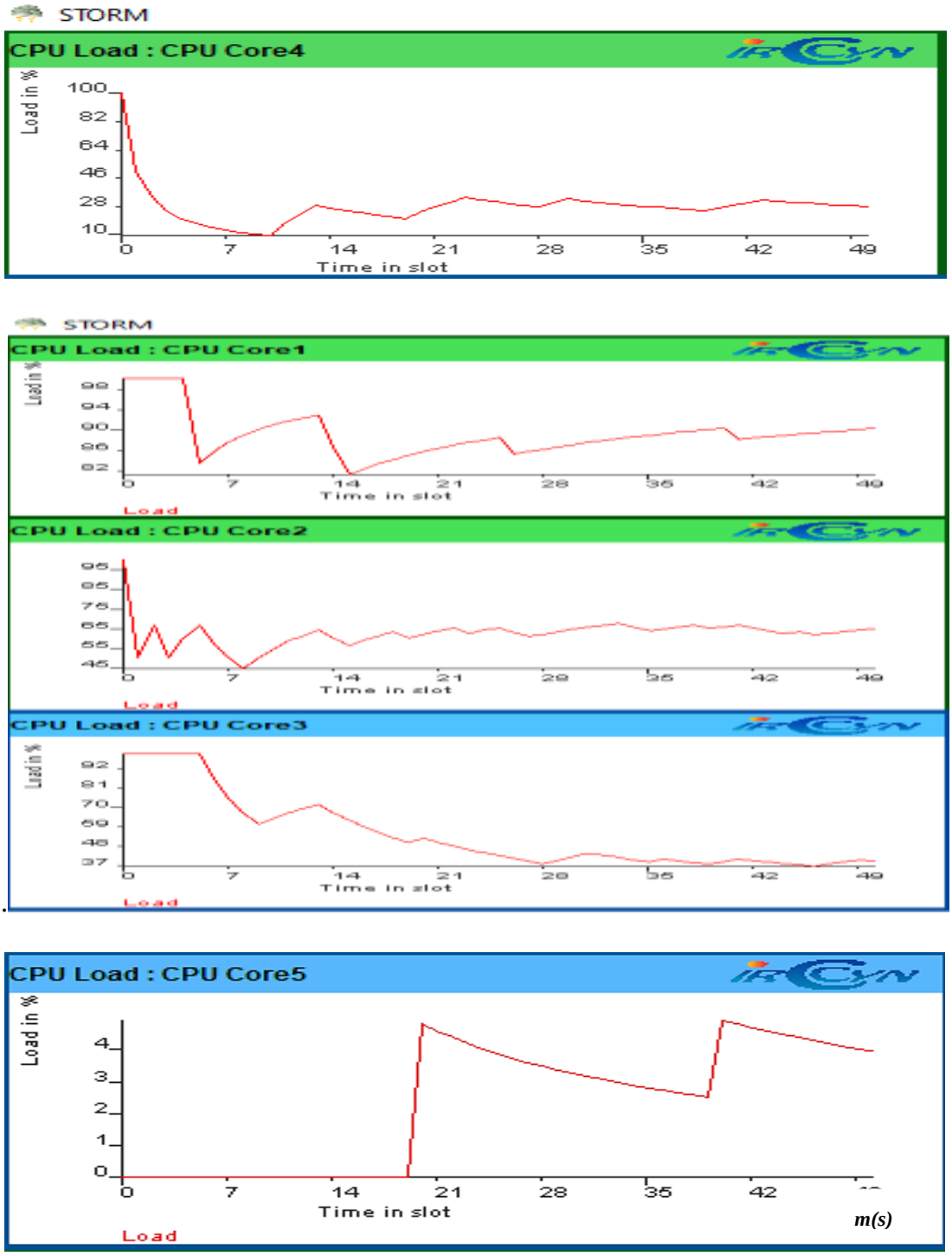


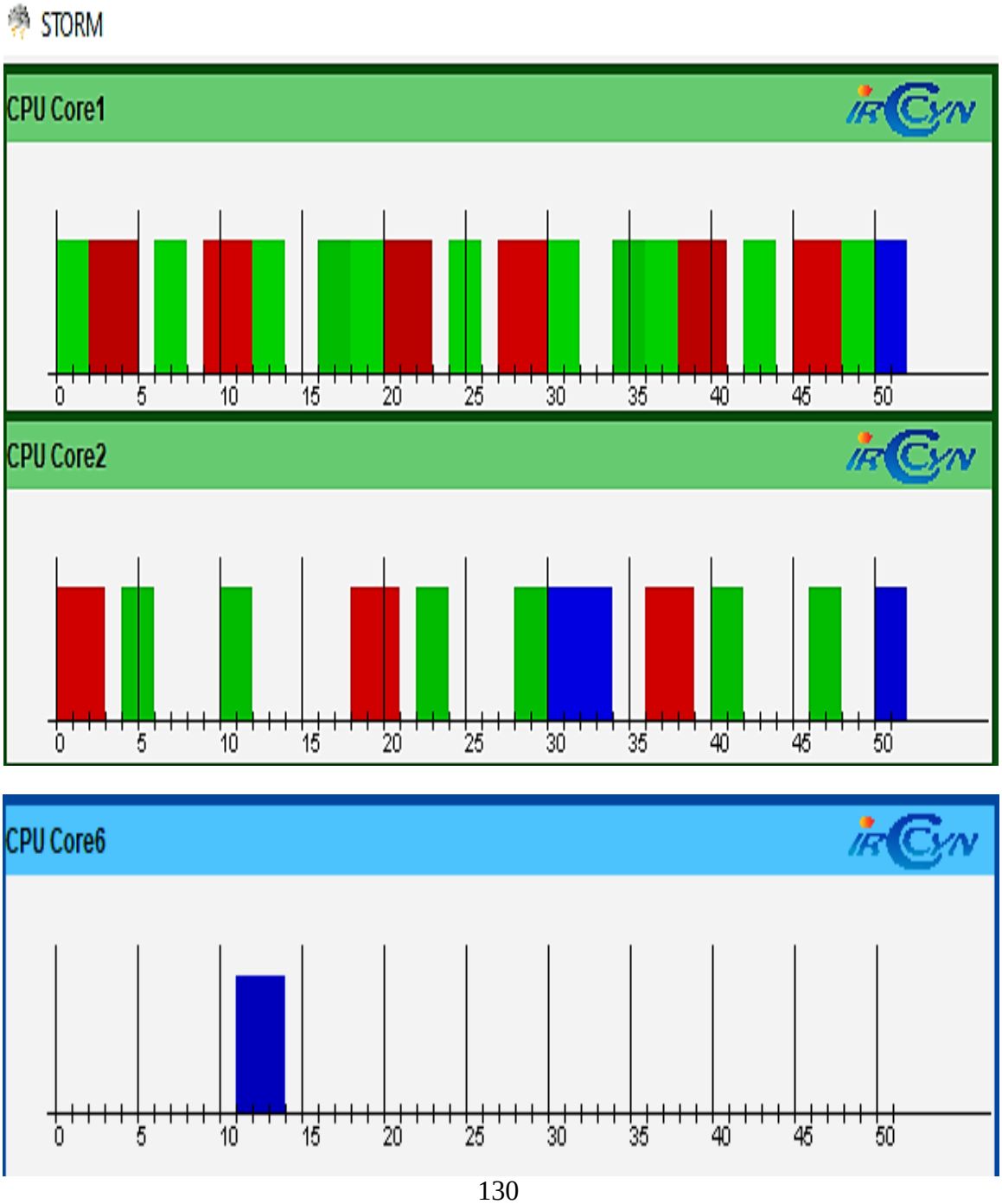
Figure 5. 28 *CPU load on core 1, 2, 3, 4 and 5 under* u_i*=38% at 624 MHz*

5.4.6 Simulation Results at u_i 45. 1 − 54%, $m = 6$

Using core configuration λ_6, s_d=1000 ms we are evaluating the results on $u_i = 38\%$ using proposed EA-EDF.

$$\forall \tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (38\%)\,,\ Core_{exe}(s)=6,\ Core_{sleep}(s)=2 \qquad (5.6)$$

When the $u_i = 50\%$ then as per the proposed configuration 6 only six cores are in running state and the rest of the two cores are in sleep mode. Below results illustrates that the entire tasks τ are scheduled on (CPU core 1, CPU core 2, CPU core 3, CPU core 4, CPU core 5 and CPU core 6) without missing their deadlines on $u_i = 50\%$.



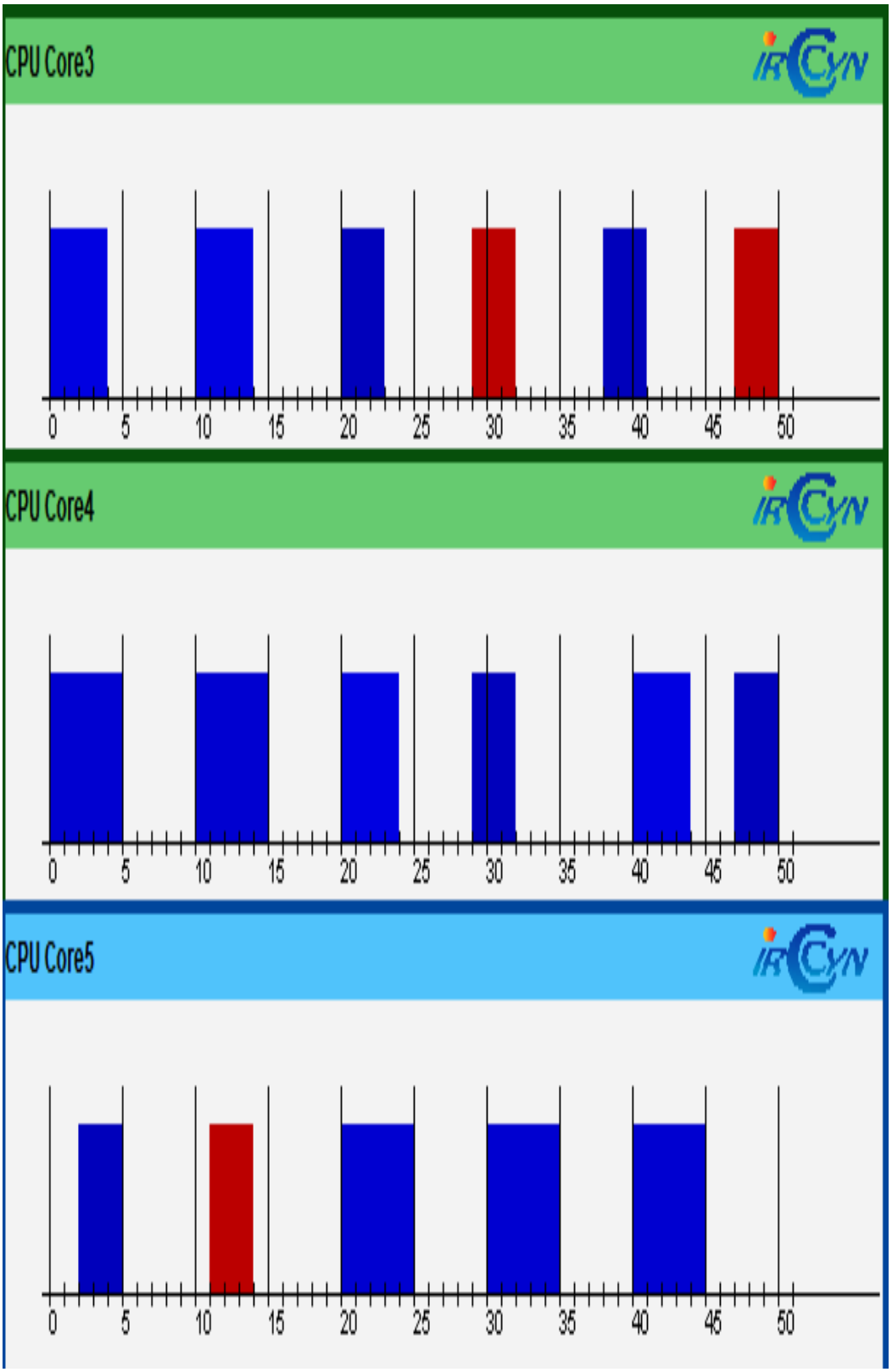


Figure 5. 29 *Scheduling of tasks using the EA-EDF technique for 6-processor configuration under* u_i*=50% at 624 MHz*

Figures 5.29 represent the exact simulation trace of a core configuration having six CPUs core with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the multiprocessor and doesn't lead to violate the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy. Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$. For validation of the simulation results using the EA-EDF scheduler the parameters for time in (ms) in the simulator are displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue color represents the ready task of $t_1$and the red color represents the ready task of t_2 while the green color represents the ready task of t_3 and so on). At CPU core 1 task $t_i = t_3$ is activated and scheduled on the CPU m_1 at time interval t= 0 till t= 2. The task $t_i = t_3$ is running and properly scheduled on CPU Core 1.

Another task $t_i \in \tau_{ready}$ queue $t_i = t_2$ is activated and scheduled on the CPU m_2 at time interval t= 0 ms till its absolute deadline on t= 3ms. $t_i = t_3$ is activated and scheduled on the CPU m_3 at time interval t= 0 ms to t= 4 ms , $t_i = t_4$ is activated and scheduled on the CPU m_4 at time interval t= 0 ms to t= 5 ms and $t_i = t_5$ is activated and scheduled on the CPU m_5 at time interval t= 2 ms to t= 5 ms because of task migration and core selection policy all the task arriving at interval t= 0 is scheduled properly without missing their deadlines and once the scheduled task is executed on the core will remain in idle mode due to dynamic power management DPM policy the CPU core 1 remains idle from time interval t= 4 ms to t= 5 ms.

CPU core 2 is moved to idle at time intervals t= 3 ms to t= 4 ms. CPU core 3 is moved to idle at time intervals t=4 ms to t= 10 ms. CPU core 4 is moved to idle at time interval t=2 ms to t= 11 ms. CPU core 5 is moved to idle at time interval t=0 ms to t= 2 ms, from t=5 ms to t= 11. CPU core 6 is moved to idle at time interval t=0 ms to t= 11 ms, from t=14 ms to t= 50. Simultaneously the task $t_i = t_1, t_2, .., t_6, t_7$ with earliest d_i deadlines are executed on $m = 1, 2, 3, 4, 5$ and 6 CPUs to avoid missing deadlines.

Simulation results show that core configuration with more than five CPUs satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration. Figure 5.30 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using gantt chart that requires a CPU to schedule.

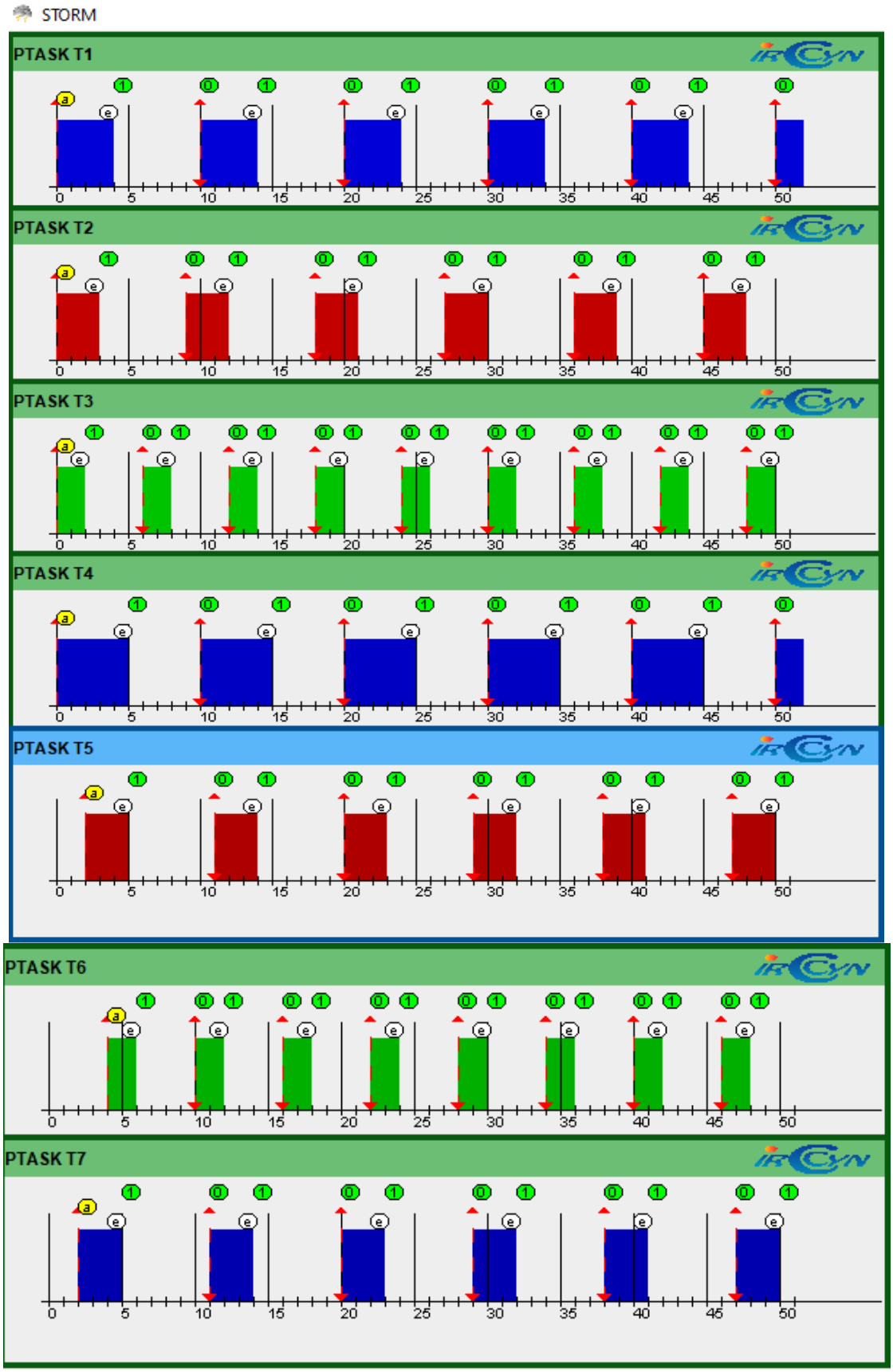


Figure 5. 30 *Gantt charts for scheduling of tasks during simulation under* u_i*=50%*

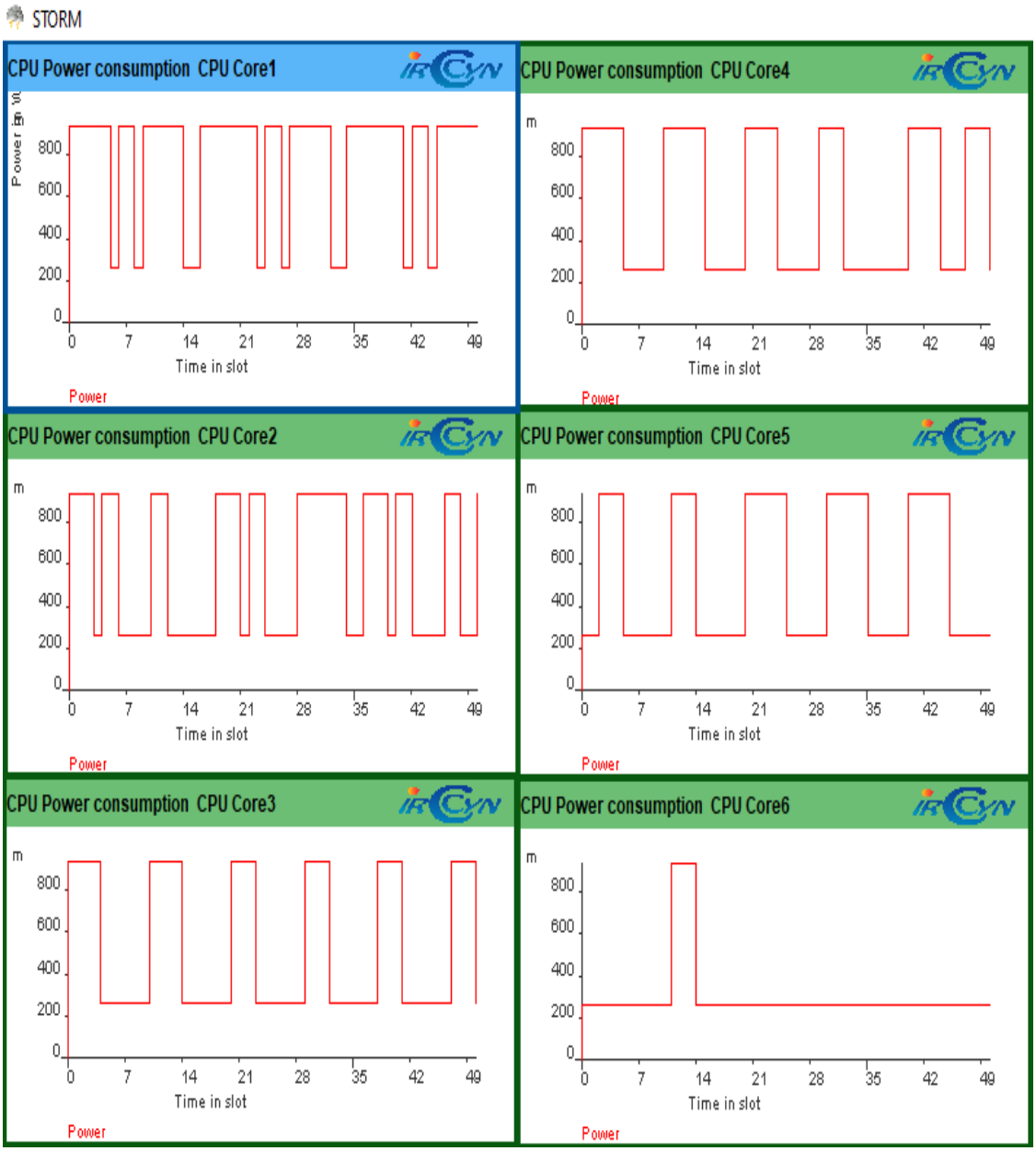


Figure 5. 31 *CPU Power Consumption on core* $p_1, p_2, ., p_6$ *under* u_i=50% *at 624 MHz*

Figure 5.31 represents the electrical power consumption for PXA- 270 multiprocessor architecture with m cores $P = \{p_1, p_2, p_3, p_4, p_5, p_6\}$. When ($u_i \geq \max threhold\ defined\ in\ \lambda_5$) then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable task load ($t_i \in \tau$) and prior shifting the next core from $P_{sleep}(\beta)$ to $P_{idle}(\Upsilon)$ in the core configuration λ_6. the consumption of cores $\{p_1, p_2, p_3, p_4, p_5, p_6\}$ } $P_{running}(\alpha)$ =0.925 W in running and consumption of $P_{idle}(\Upsilon)$ 0.26 W in idle mode.

At CPU cores $\{p_3, p_4, p_5, p_6\}$ it's visible that DPM shifts the core to a lower power consumption state (Υ) by increasing the low power $P_{idle}(\Upsilon)$ idle duration specifically at

core $\{p_3, p_4, p_5, p_6\}$ is because EA-EDF uses task migration that schedules a task on the core in such an efficient way that less no of processing cores but due to an increase in task load the increase in energy slightly increases as compared to the λ_5 the $P_{running}(\alpha) = 6$ more cores are required to execute the task but still processor reduces the overall energy consumption by keeping $P_{sleep}(\beta) = 2$ sleep mode using core configuration λ_6.

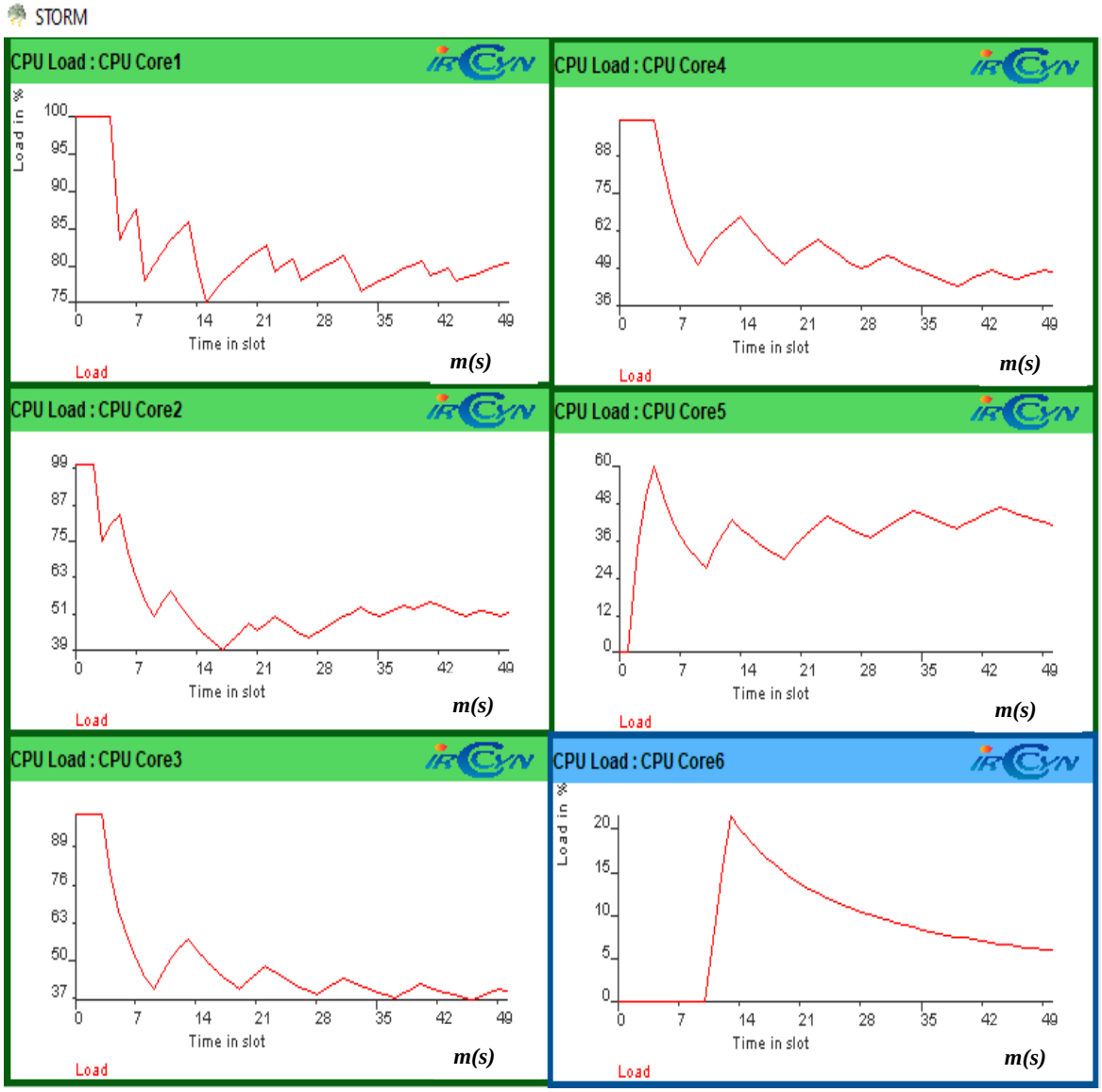


Figure 5. 32 *CPU load on the core* $p_1, p_2, \ldots, p_6$ *under* u_i=50% *at 624 MHz*

Figure 5.32 represents the CPU load shows the scheduling tasks on various cores. The higher peak of the curve shows that multiple tasks $t_i \in \tau$ requires the CPU time to schedule without missing their d_i increases the utilization u_i. The fall of the curve

shows that DPM enhances the idle duration to maintain a low power mode that results in lower energy consumption using EA-EDF.

5.4.7 Simulation Results at u_i 54. 1 − 62. 5%, $m = 7$

Using core configuration λ_7, s_d=1000 ms we are evaluating the results on $u_i = 55\%$ using proposed EA-EDF.

$$\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} = (55\%)\,,\ Core_{exe}(s)=7,\ Core_{sleep}(s)=1 \tag{5.7}$$

When the $u_i = 55\%$ then as per λ_6, only 7 cores are in running state and one of the cores is in sleep mode. Figures 5.33 represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using Gantt chart. To validate the simulation results of the EA-EDF scheduler for this the parameters for time in (ms) in the simulator that is displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors. t_i activated on CPU m_1 ,m_2, m_3 and m_4 simultaneously at time interval t= 0 due to high task load. $t_i \in \tau$ starts migration between the different τ_{ready} the queue to have always m CPUs and the task $t_i = t_1, \ldots t_n$; n > 0 with the earliest d_i deadlines are ready to be executed on $m = 6$ CPUs.

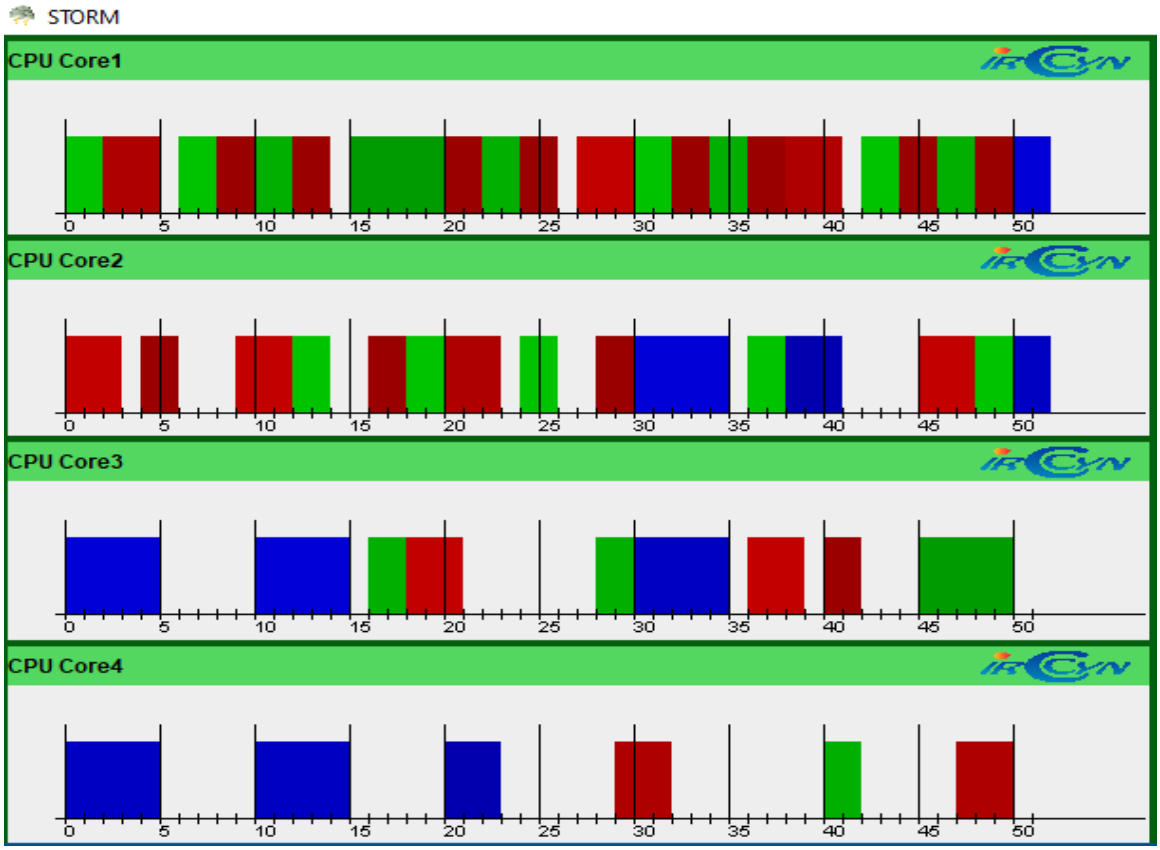


Figure 5. 33 *Scheduling of task on core 1, 2, 3 and 4 under u_i=55% at 624 MHz*

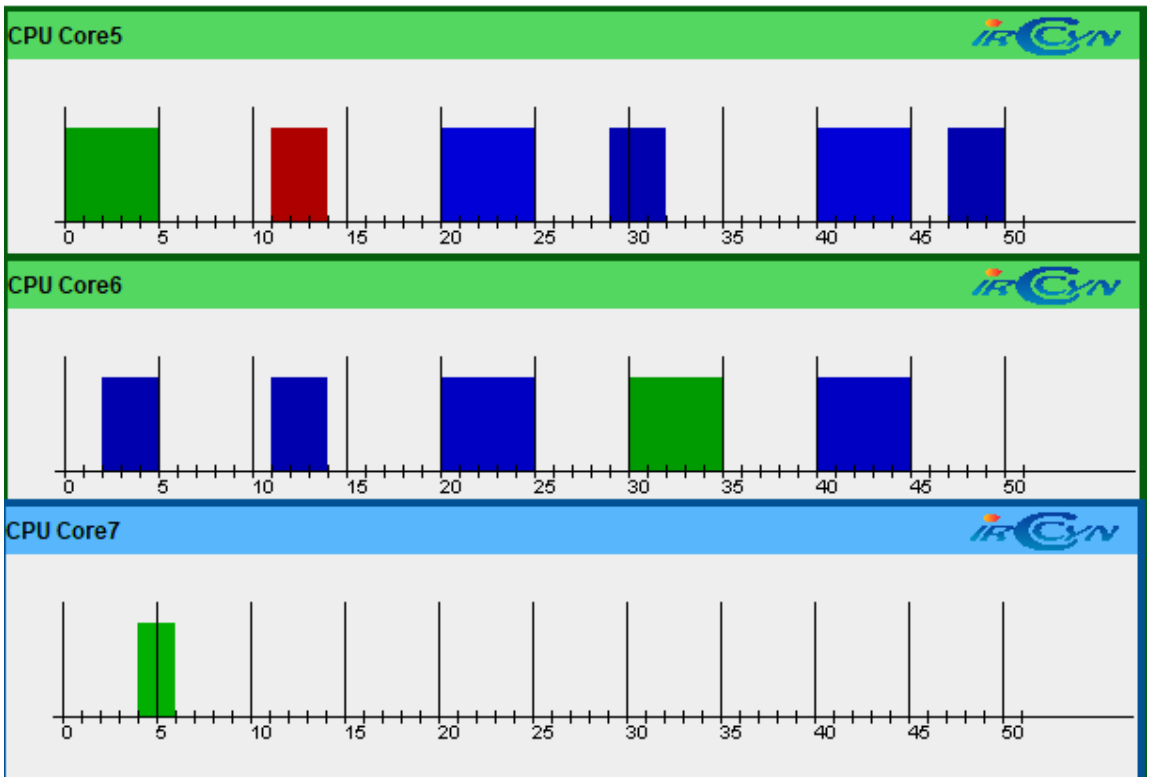


Figure 5. 34 *Scheduling of tasks using the EA-EDF technique for 7-processor configuration under* u_i*=55% at 624 MHz*

Figures 5.34 represent the exact simulation trace of core configuration having seven CPUs core with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the multiprocessor and doesn't lead to violate the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy. Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$. For validation of the simulation results using the EA-EDF scheduler the parameters for time in (ms) in the simulator are displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue color represents the ready task of $t_1$and the red color represents the ready task of t_2 while the green color represents the ready task of t_3 and so on). At CPU core 1 task $t_i = t_3$ is activated and scheduled on the CPU m_1 at time interval t= 0 till t= 2. The task $t_i = t_3$ is running and properly scheduled on CPU Core 1. Another task $t_i \in \tau_{ready}$ queue $t_i = t_2$ is activated and scheduled on the CPU m_2 at time interval t= 0 ms till its absolute deadline on t= 4ms. $t_i = t_1$ is activated and scheduled on the CPU m_3 at time interval t= 0 ms to t= 4 ms , $t_i = t_4$ is activated and scheduled on the CPU m_4 at time interval t= 0 ms to t= 5 ms and the second set of $t_i = t_1$ is activated and scheduled on the CPU m_5 at time interval t= 0 ms to t= 5 ms because of task migration and core selection policy all the

task arriving at interval t= 2 is scheduled properly without missing their deadlines on CPU m_6 and a task is migrated to the CPU m_6 at time interval t=4ms to t=6 ms once the scheduled task is executed on the core will remain in idle mode due to dynamic power management DPM policy the CPU cores remain idle when no task is scheduled. Simultaneously the task $t_i = t_1, t_2, t_3, t_4, t_5, , t_6, t_7, t_8, t_9$ with earliest d_i deadlines are executed on $m = 1, 2,3,4,5,6$ and 7 CPUs to avoid missing deadlines.

Simulation results show that core configuration with more than six CPUs satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration. Figures 5.35 and 5.36 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using gantt chart that requires a CPU to schedule.

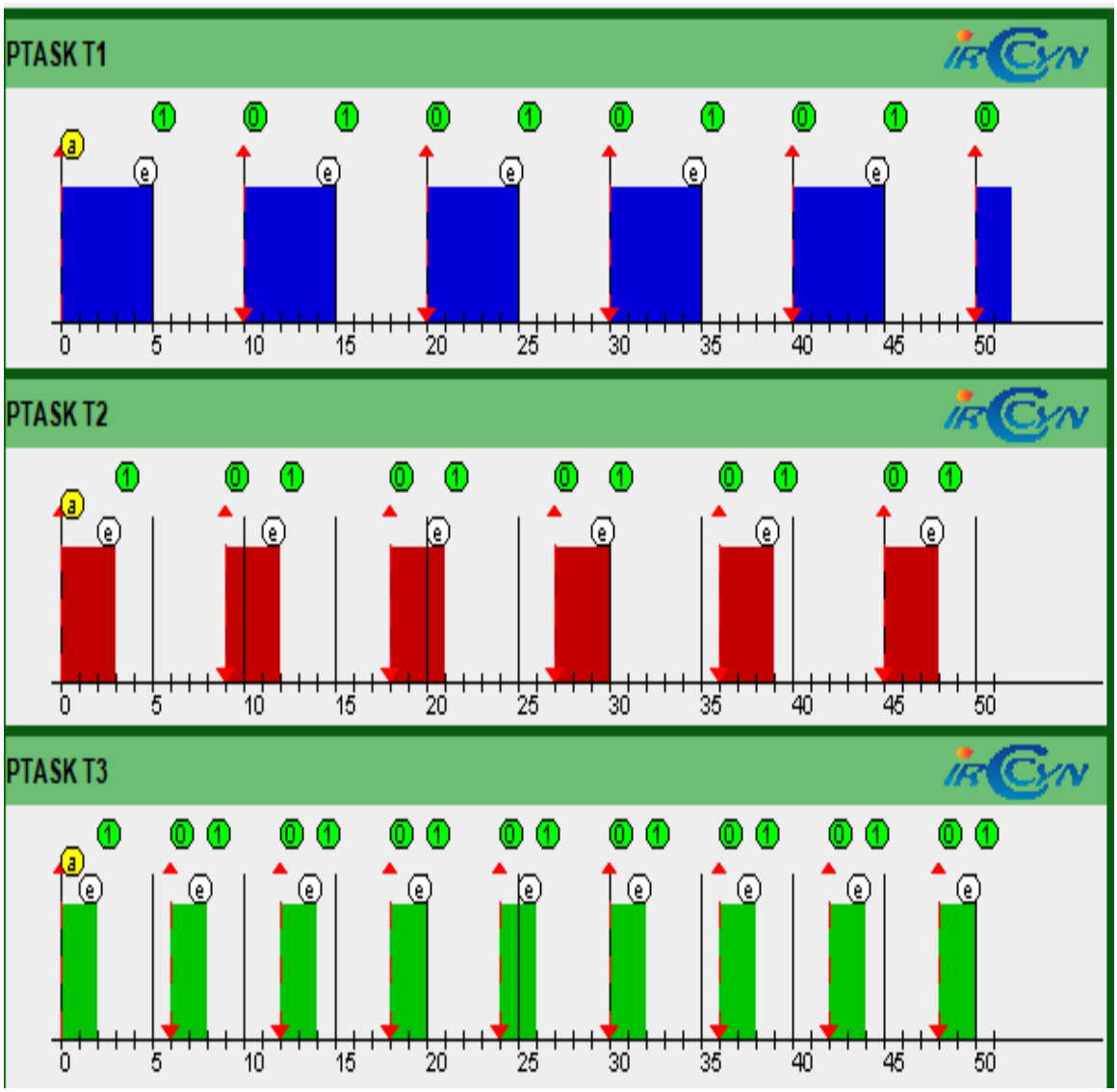


Figure 5. 35 *Gantt charts for scheduling of tasks t_1, t_2, t_3 during simulation under u_i=55%*

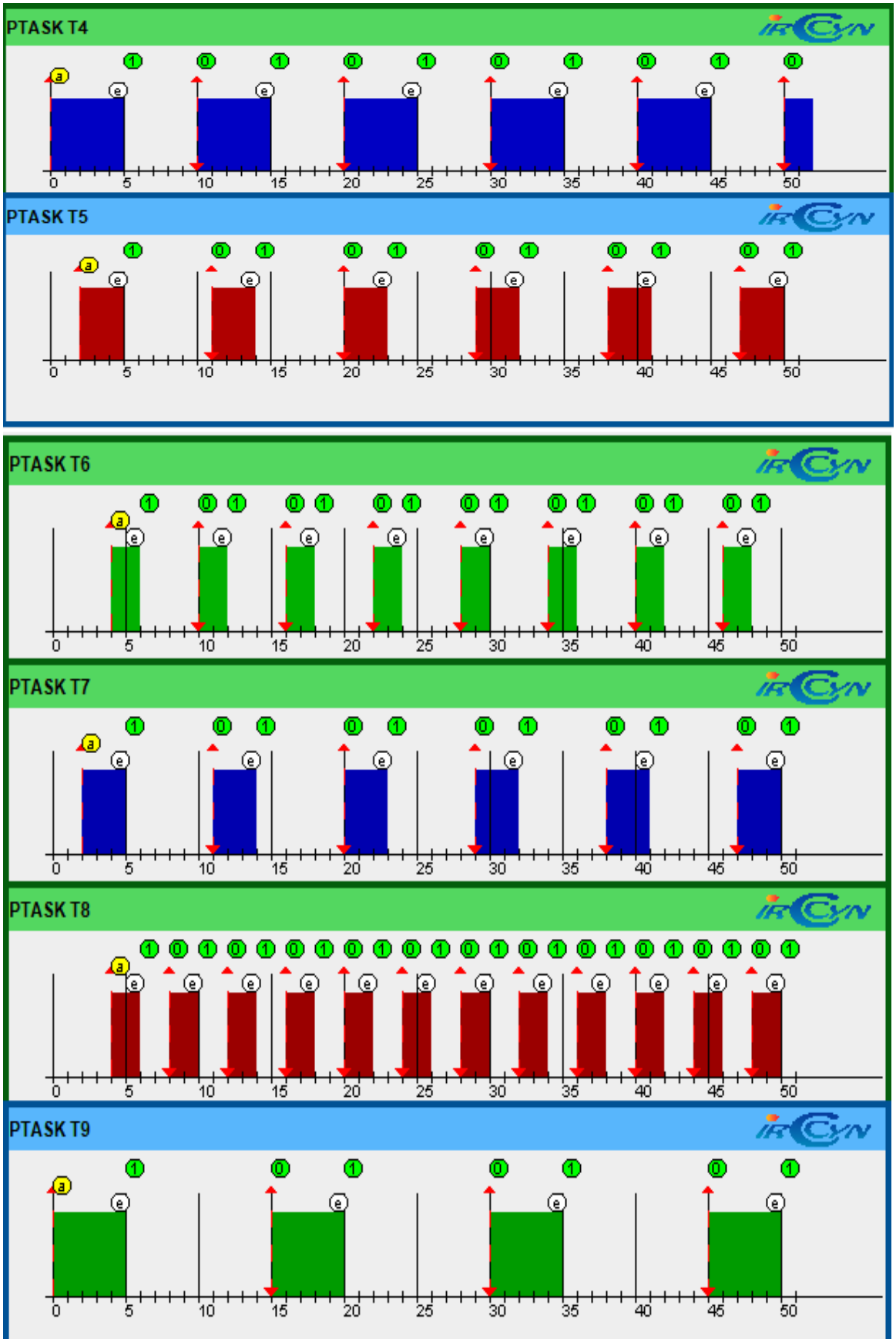


Figure 5. 36 *Gantt charts for scheduling of tasks $t_4, t_5, \ldots t_9$ during simulation under u_i=55%*

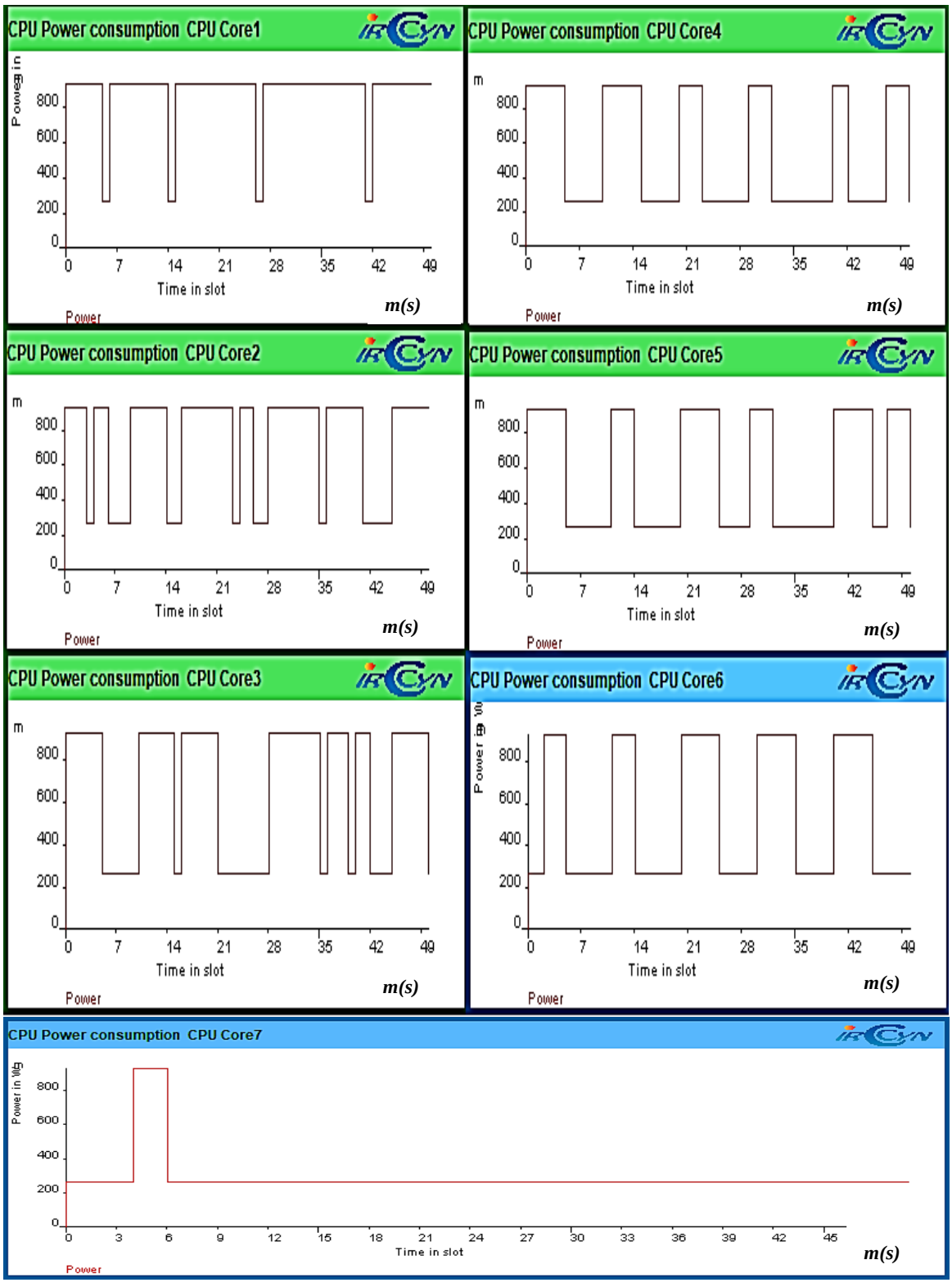


Figure 5. 37 *CPU Power Consumption on core* $p_1, p_2, \ldots, p_7$ *under* u_i*=55% at 624 MHz*

When $(u_i \geq \max threhold\ defined\ in\ \lambda_6)$ then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable

task load ($t_i \in \tau$) and prior shifting the next core from $P_{sleep}(\beta)$ to P_{idle} (Υ) in the core configuration λ_7. Figure 5.37 represents the electrical power consumption for PXA- 270 multiprocessor architecture with m cores $P = \{p_1, p_2, .., p_6, p_7\}$ under u_i=55% at 624 MHz the consumption of cores $\{p_1, p_2, .., p_7\}$. $P_{running}$ (α) =0.925 W in running and consumption of P_{idle} (Υ) 0.26 W in idle mode. At CPU cores $\{p_1, p_7\}$ it's visible that DPM shifts all the core to a lower power consumption state (Υ) by increasing the low power P_{idle} (Υ) idle duration specifically at core $\{p_1, p_7\}$ is because EA-EDF uses task migration that schedules a task on the core $P_{running}$ (α) = 7 executing the task and more cores remain in $P_{running}(\alpha)$ state that causes the processor to increase consumption of energy by keeping less no of $P_{sleep}(\beta) = 1$ sleep mode using core configuration λ_7.

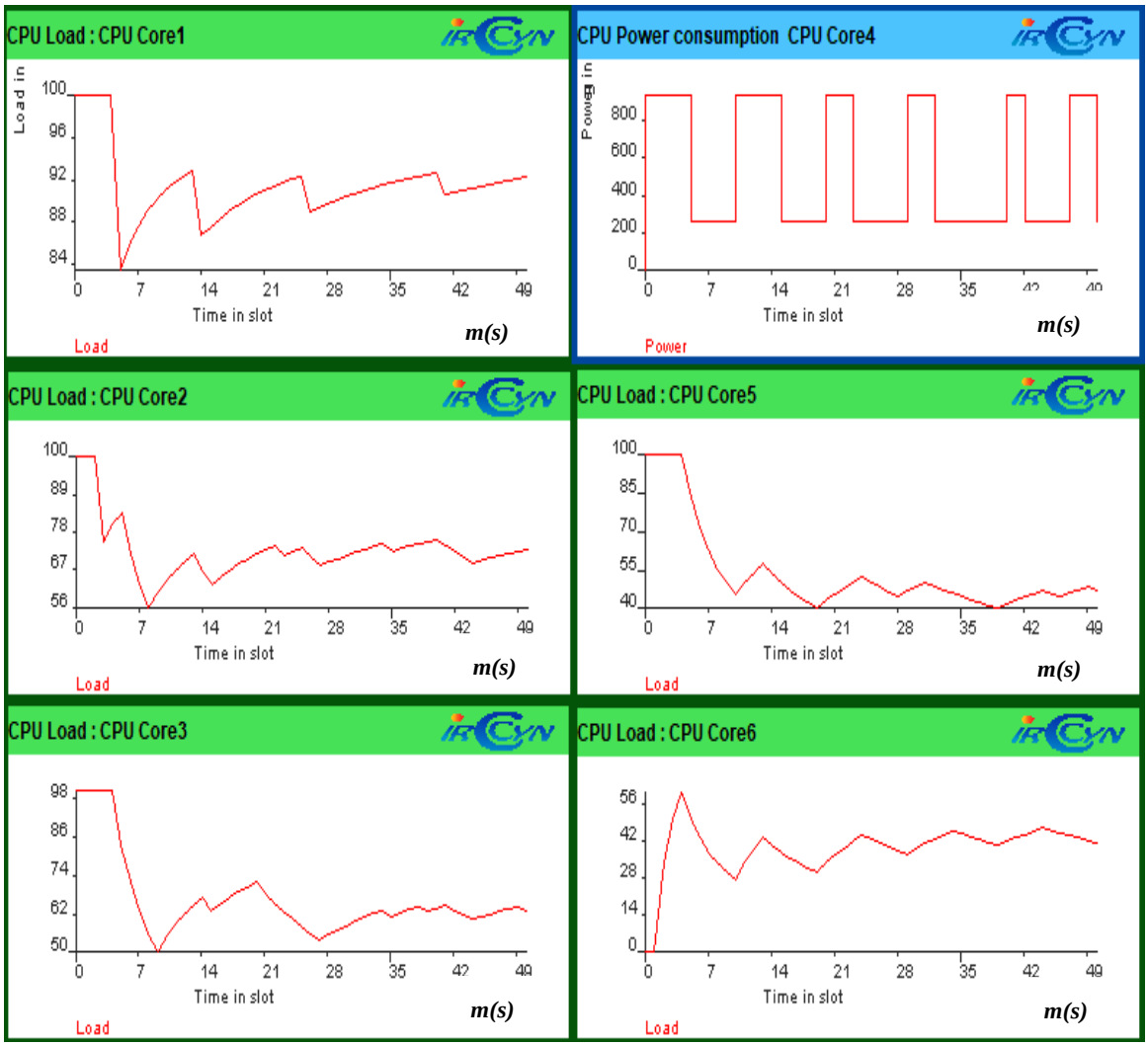


Figure 5. 38 *CPU load on the core* $p_1, p_2, \ldots, p_7$ *under* u_i=55% *at 624 MHz*

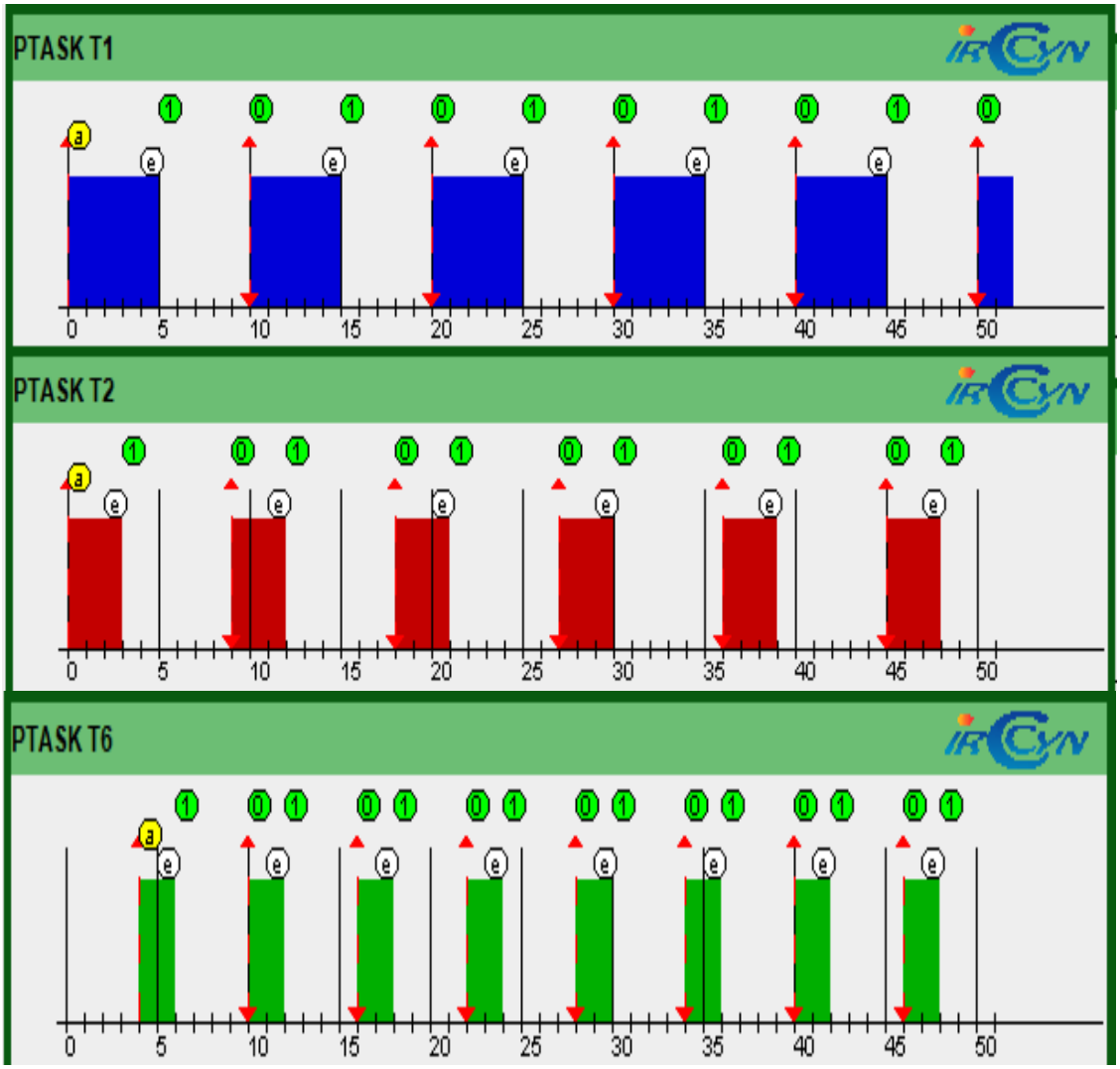
Figure 5.38 represents the CPU load shows the scheduling tasks on various cores. The higher peak of the curve shows that multiple tasks $t_i \in \tau$ requires the CPU time to schedule without missing their d_i increases the utilization u_i. The fall of the curve shows that DPM enhances the idle duration to maintain a low power mode that results in lower energy consumption using EA-EDF.

5.4.8 Simulation Results at $u_i > 62.5\%, m = 8$

Using configuration λ_8, s_d=1000ms we are evaluating the results on $u_i > 62.5\%$ using proposed EA-EDF.

$$\forall \tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} > (62.5\%)\,,\ Core_{exe}(s)=8,\ Core_{sleep}(s)=0 \qquad (5.8)$$

When the $u_i > 62.5\%$ than as per the proposed configuration 8 all the cores are in running state in an octa-core processor. Below results illustrates that the entire tasks τ are scheduled on various cores without missing their deadlines on $u_i > 62.5\%$.



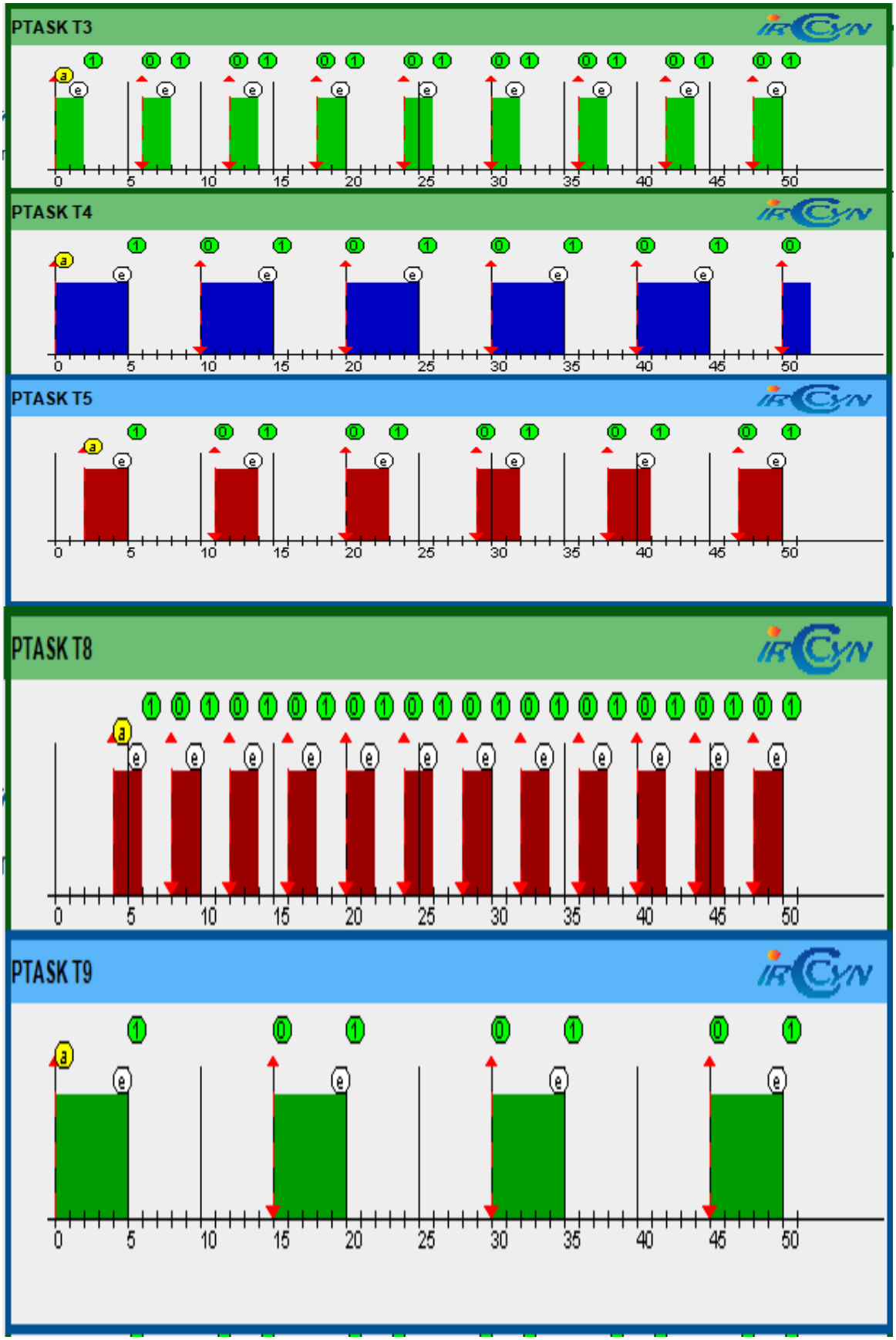


Figure 5. 39 *Gantt charts for scheduling of tasks during simulation under* $u_i > 62.5\%$

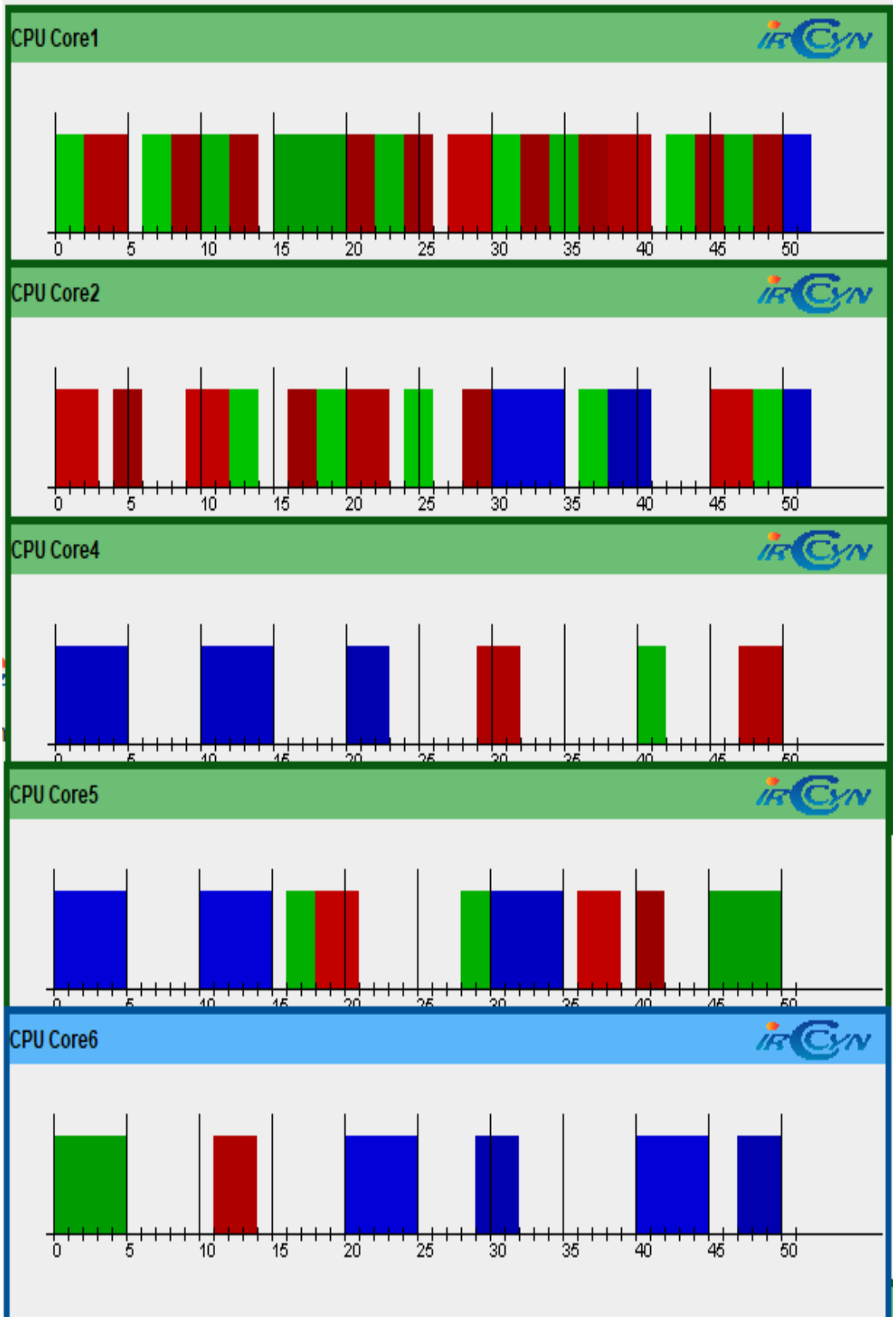


Figure 5. 40 *Scheduling of tasks using the EA-EDF technique for 8-processor configuration* $(p_1, p_2, p_4, p_5, p_6)$ u_i*=62.5% at 624 MHz*

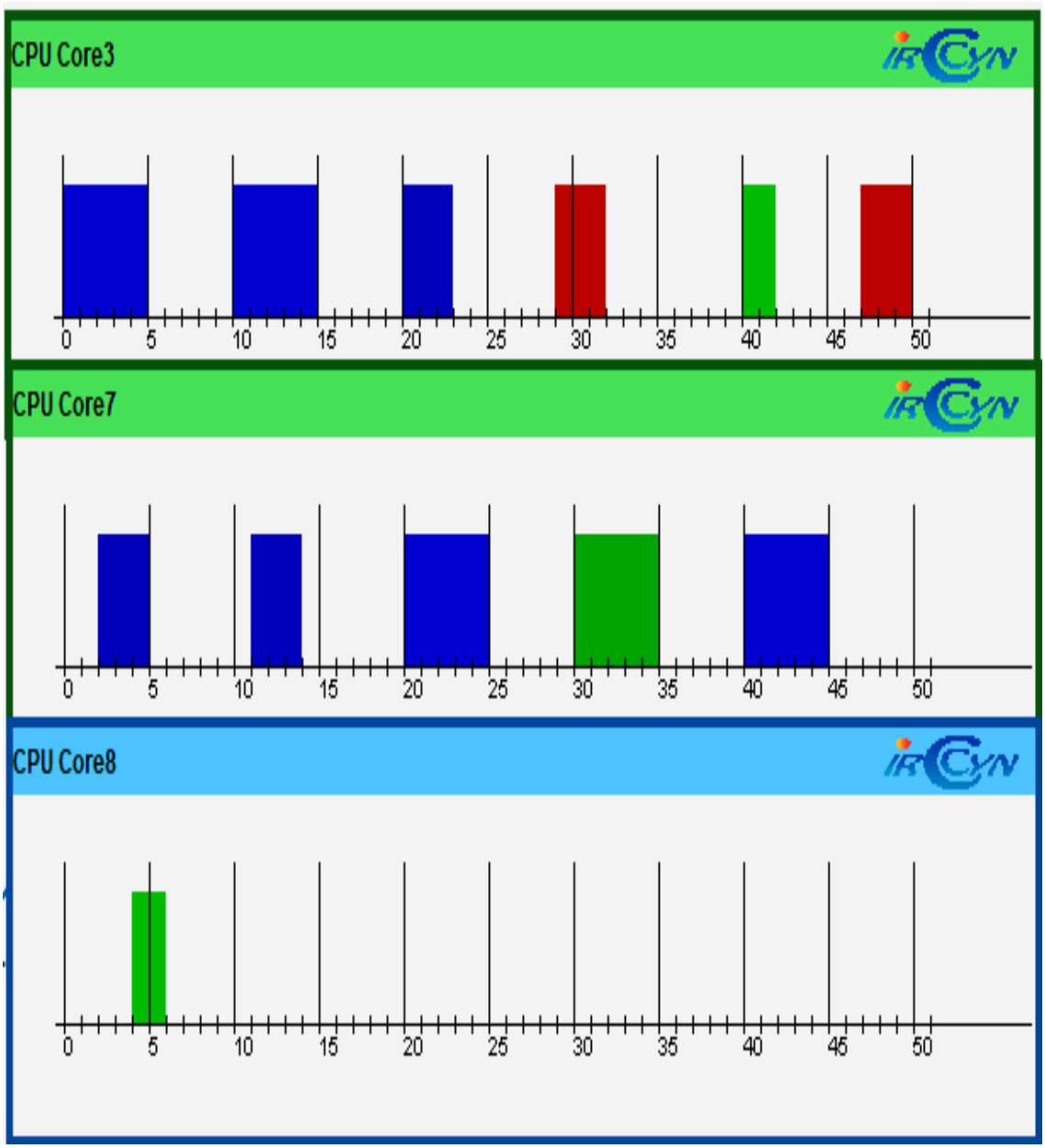


Figure 5. 41 *Scheduling of tasks using the EA-EDF technique for 8-processor configuration (p_3, p_7, p_8) u_i=62.5% at 624 MHz*

Figures 5.39 and Figures 5.40 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using gantt chart that requires a CPU to schedule.

Figures 5.41 represent the exact simulation trace of a core configuration having seven CPUs core with the time interval 0-50 ms. All the task are properly scheduled on the two cores of the multiprocessor and doesn't lead to violate the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy. Gantt chart represents the arrival, activation, blocking, deadline, release and scheduling

of different tasks $t_i \in \tau$. For validation of the simulation results using the EA-EDF scheduler the parameters for time in (ms) in the simulator are displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors (blue color represents the ready task of $t_1$and the red color represents the ready task of t_2 while the green color represents the ready task of t_3 and so on).

At CPU core 1 task $t_i = t_3$ is activated and scheduled on the CPU m_1 at time interval t= 0 till t= 2. The task $t_i = t_3$ is running and properly scheduled on CPU Core 1. Another task $t_i \in \tau_{ready}$ queue $t_i = t_2$ is activated and scheduled on the CPU m_2 at time interval t= 0 ms till its absolute deadline on t= 3ms. $t_i = t_1$ is activated and scheduled on the CPU m_4 at time interval t= 0 ms to t= 5 ms , $t_i = t_4$ is activated and scheduled on the CPU m_5 at time interval t= 0 ms to t= 5 ms and the second set of $t_i = t_1$ is activated and scheduled on the CPU m_3 at time interval t= 0 ms to t= 5 ms because of task migration and core selection policy all the task arriving at interval t= 2 to t=5 is scheduled properly without missing their deadlines on CPU m_6 and a task is migrated to the CPU m_7 at time interval t=2ms to t=5 ms, and a task is migrated to the CPU m_8 at time interval t=4ms to t=6 ms once the scheduled task is executed on the core will remain in idle mode due to dynamic power management DPM policy the CPU cores remain idle when no task is scheduled. Simultaneously the task $t_i = t_1, t_2, \ldots, t_8, t_9$ with earliest d_i deadlines are executed on $m = 1, 2, 3, 4, 5, 6, 7$ and 8 CPUs to avoid missing deadlines. the task $t_i = t_1, \ldots t_n$; $u > 62.5\%$ task with the earliest d_i deadlines are ready to be executed on $m = 8$ CPUs and behave similar to the other global techniques. Simulation results show that core configuration with more than seven CPUs satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration.

When ($u_i > \max threhold\ defined\ in\ \lambda_7$) then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable task load ($\boldsymbol{t_i \in \tau}$) and prior shifting the next core from $P_{sleep}(\beta)$ to P_{idle} (ϒ) in the core configuration λ_8. Figure 5.42 represents the electrical power consumption for PXA- 270 multiprocessor architecture with m cores $P = \{p_1, p_2, \ldots, p_8\}$ under u_i>62.5% at 624 MHz the consumption of cores $\{p_1, p_2, \ldots, p_8\}$ $P_{running}$ (α) =0.925 W in running and consumption of P_{idle} (ϒ) 0.26 W in idle mode.

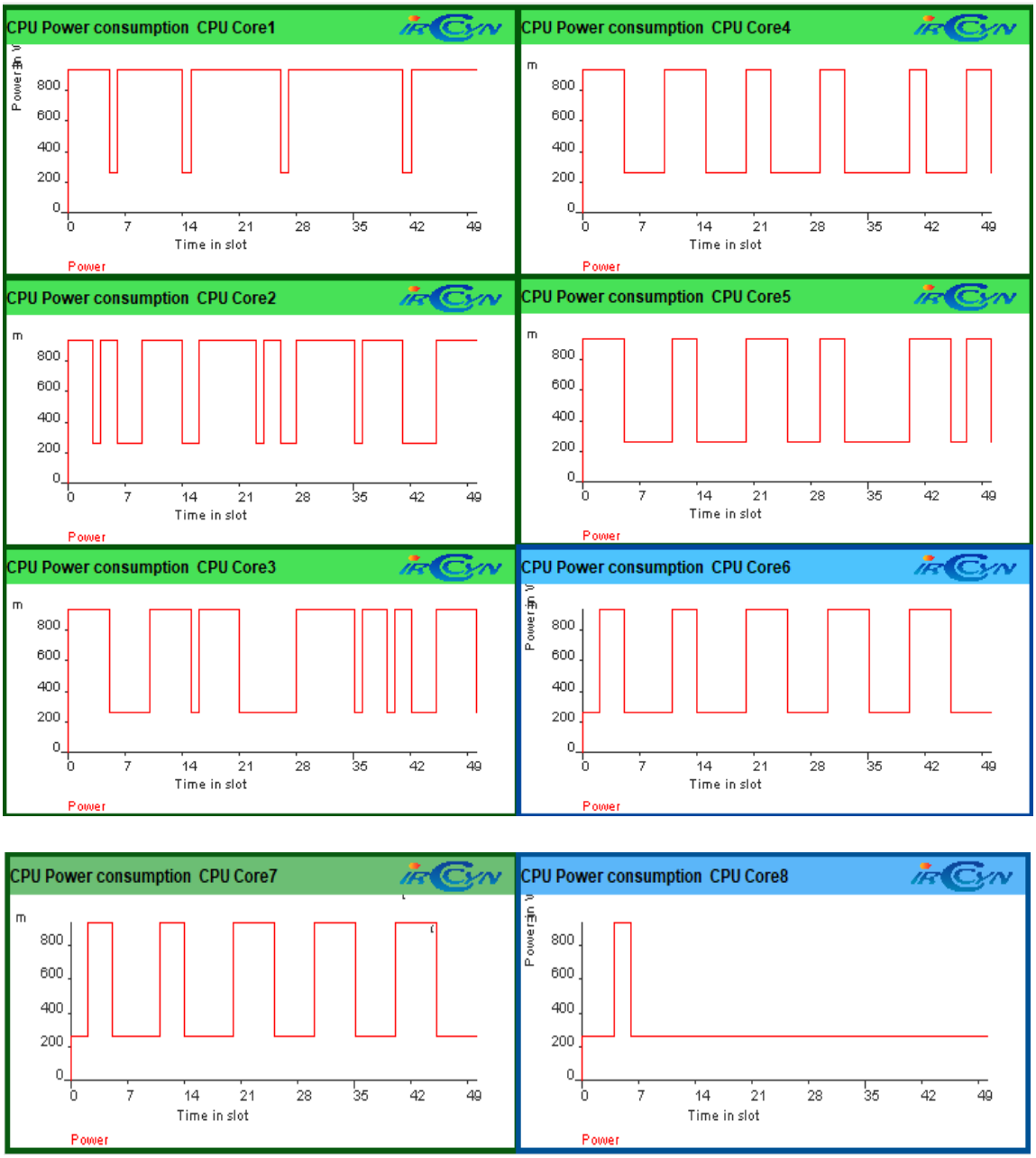


Figure 5. 42 *CPU Power Consumption on core* $P = \{p_1, p_2, \ldots\ldots, p_8\}$ *under* u_i>62.5% *at 624 MHz*

At CPU cores $\{p_1, p_7\}$ it's visible that all the task are meeting their deadlines and properly scheduled and DPM tries to shift the core to a lower energy and power dissipation mode but the (Υ) idle duration is very minimal and EA-EDF behaves similar to the G-EDF because all the core $P_{running}\ (\alpha) = 8$ are executing the task and no cores remain in $P_{idle}\ (\Upsilon)$ state that causes the processor to increase the consumption of energy like other techniques.

Figure 5. 43 *CPU load on the core* $p_1, p_2, \ldots\ldots, p_8$ *under* u_i*>62.5% at 624 MHz*

While Figure 5.43 represents the CPU load on various cores that are scheduling the task. The higher peak of the curve shows that multiple tasks $t_i \in \tau$ requires the CPU time to schedule without missing their d_i increases the utilization u_i. Each core has a variable load increasing with an increase in utilization and decreasing as DPM enabled the idle duration for low power mode.

The fall of the curve shows that DPM enhances the idle duration to maintain a low power mode that results in lowering the energy consumption using EA-EDF.

5.4.9 Simulation Results H.264 video decoder application

H.264 video decoder application is evaluated on simulation duration s_d=*1000* ms using proposed EA.EDF scheduling technique for multiprocessor architecture with $m = 8$ cores with P $= \{p_1, p_2, \ldots., p_8\}$ and a total of *n=7* tasks $t_i \in \tau$ required resources to schedule on a core configuration ($\lambda_7 \in \lambda_n$).

Each task $t_i \in \tau$ consists of a periodic task set TASKS $t_i \in \tau$ represented as $\tau = (t_1, t_2, t_3, \ldots..t_n)$ and its parameters $a_{i,}, c_i, d_{i,} p_{i,} p_r$ for $(t_1, t_2, \ldots\ldots, t_7)$ are defined in chapter 4: section 4.6.1

From simulation results, it is observed that only 5 cores are enough to execute all the 7 tasks of the H.264 multimedia decoder application by using core in the configuration λ_5.

$$\forall\tau \rightarrow u_\tau = \sum_{k=1}^{n} \frac{c_a}{p_a} > (36.1 - 45\%)\,,\ Core_{exe}(s)=5,\ Core_{sleep}(s)=3 \qquad (5.9)$$

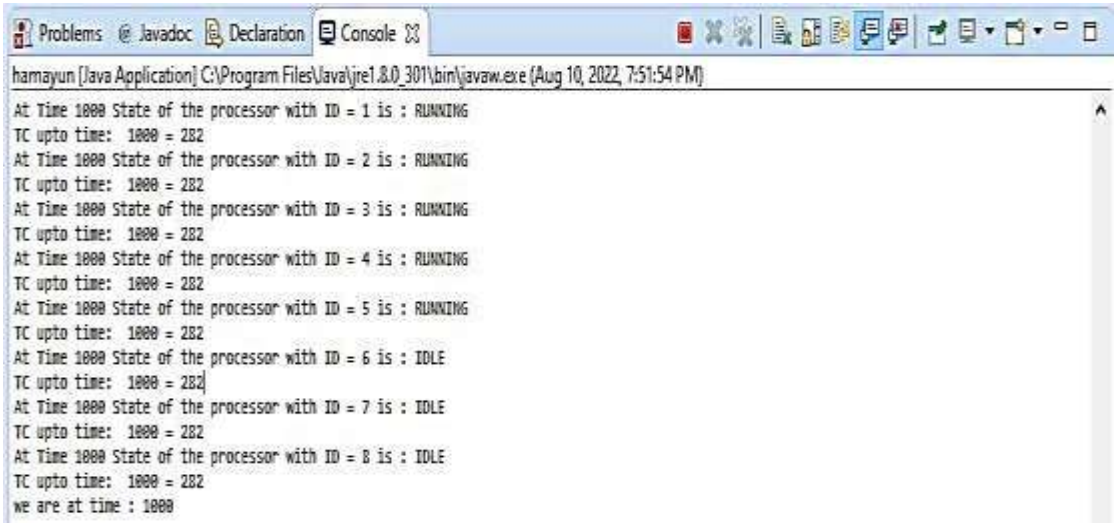


Figure 5. 44 *shows the real-time simulation of 5 running & 3 idle processor*

Based on the simulation results as shown in Figure 5.46 it is estimated that MPSoC with 5 running processors in the core configuration λ_5 is more suitable to schedule all 7 tasks of the H.264 multimedia decoder application.

Figures 5.45 represent the arrival, activation, blocking, deadline, release and scheduling of different tasks $t_i \in \tau$ using the gantt chart that requires a CPU to schedule.

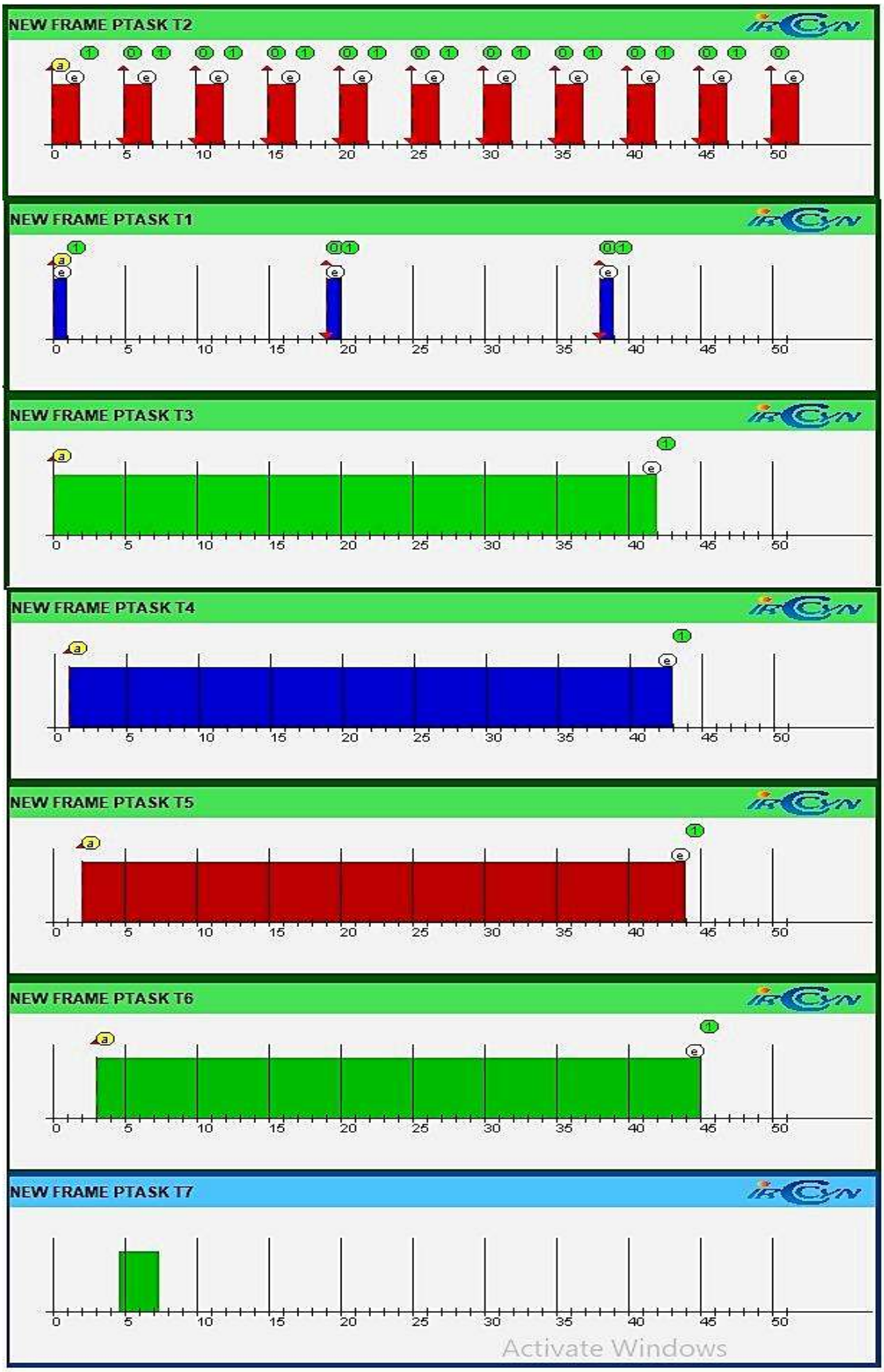


Figure 5. 45 *Scheduling of H.264 multimedia decoder application task set using proposed EA-EDF technique at 624 MHz*

Figures 5.46 represent the simulation trace of a core configuration having five cores with a time interval 0-50 ms. All the task are properly scheduled on five cores of the multiprocessor and doesn't lead to violating the deadline for the execution of the ready tasks from the task set $\forall t_i \in \tau$ using the proposed core configuration policy.

The vertical Y-axis represents the schedules and time interval through the STORM is displayed in (X-axis) units of 50 ms each $t_i \in \tau$ is displayed using different colors, the light blue color represents the ready task of t_1, the dark blue color represents the ready task of t_4, light red color represents the ready task of t_2 , dark red color represents the ready task of t_5 and green color represents the ready task of t_3, t_6, t_7 etc.

At CPU core 1 new frame task $t_i = t_2$ is activated and scheduled on the CPU m_1 at time interval t= 0 till t= 2. The task $t_i = t_2$ is running and properly scheduled on CPU Core 1 until its deadline at t= 2. Another task $t_i = t_5 \in \tau_{ready}$ is activated and scheduled on the CPU m_1 at time interval t= 0 ms till its absolute deadline on t= 44 ms.

$t_i = t_1$ is activated and scheduled on the CPU m_2 at time interval t= 0 ms to t= 1 ms, $t_i = t_4$ (new frame) is activated and scheduled on the CPU m_2 at time interval t= 1 ms to t= 43 ms. Another task of $t_i = t_3$ is activated and scheduled on the CPU m_3 at time interval t= 0 ms to t= 45 ms another new frame task $t_i = t_6$ arrive at the same time interval t=2 schedule on m_4 because of task migration and core selection policy, all the task arriving at interval t= 2 to t= 45 ms is scheduled properly without missing their deadlines on the CPU m_4 because CPU m_3 is already occupied with the task in the same time frame.

Similarly some new frames of task $t_i = t_2$ arrive at interval t= 5 to t= 7 till 20 ms and $t_i = t_1$ arrive at t=19 to t=20 and t=37 to t=38 but at that time all the four CPUs m_1, m_2, m_3 and m_4 are completely occupied so the task migration policy starts migrating these new frames to the core m_5 and the rest of the 3 cores including (m_6, m_7, m_8) in the system remains idle or in sleep mode, due to DPM policy, the CPU cores remain idle when no task is scheduled.

Simulation results show that task $t_i = t_1, t_2, t_3, \ldots, t_7$ with earliest d_i deadlines are executed on $m = 1, 2, 3, 4,$ and 5 CPUs to guarantee meeting deadlines. Core configuration with more than five CPUs satisfied the required quality of service QoS and performance as these systems are based on energy consumption results in design exploration.

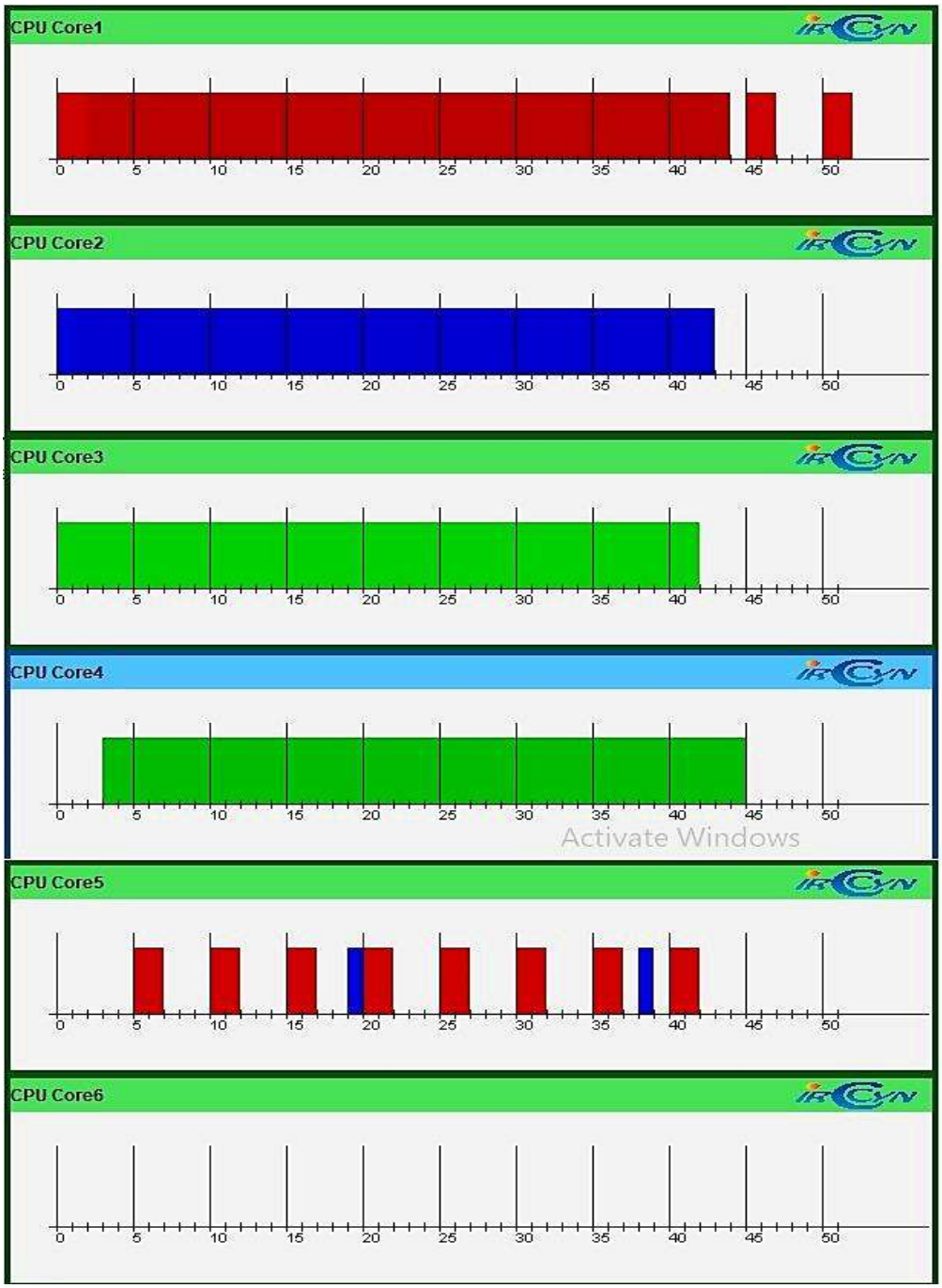


Figure 5. 46 *Scheduling of H.264 multimedia decoder application tasks using the EA-EDF on configuration based o at 624 MHz*

Figure 5.47 represents the electrical power consumption for INTEL PXA-270 multiprocessor architecture with m cores $P = \{p_1, p_2, p_3, p_4, p_5\}$ under H.264 multimedia decoder application at 624 MHz. When ($u_i \geq max\ threhold\ defined\ in\ \lambda_4$) then the task migration policy selects a suitable core configuration that meets the scheduling and u_i requirement of migratable task load ($t_i \in \tau$) and prior shifting to the next core from $P_{sleep}(\beta)$ to $P_{idle}(\Upsilon)$ in the core

configuration λ_5 the consumption of cores {1,2,3,4,5} $P_{running}(\alpha)$ =0.925 W in running and consumption of $P_{idle}(\Upsilon)$ 0.26 W in idle mode. Due to task migration schedules a task on the cores efficiently and guarantees to meet the deadlines $P_{running}(\alpha) = 5$ executing the task and fewer cores remain in $P_{sleep}(\beta)$ or idle state that causes the multiprocessor to conserve the total energy consumption by keeping $P_{sleep}(\beta) = 3$ sleep mode using core configuration λ_5.

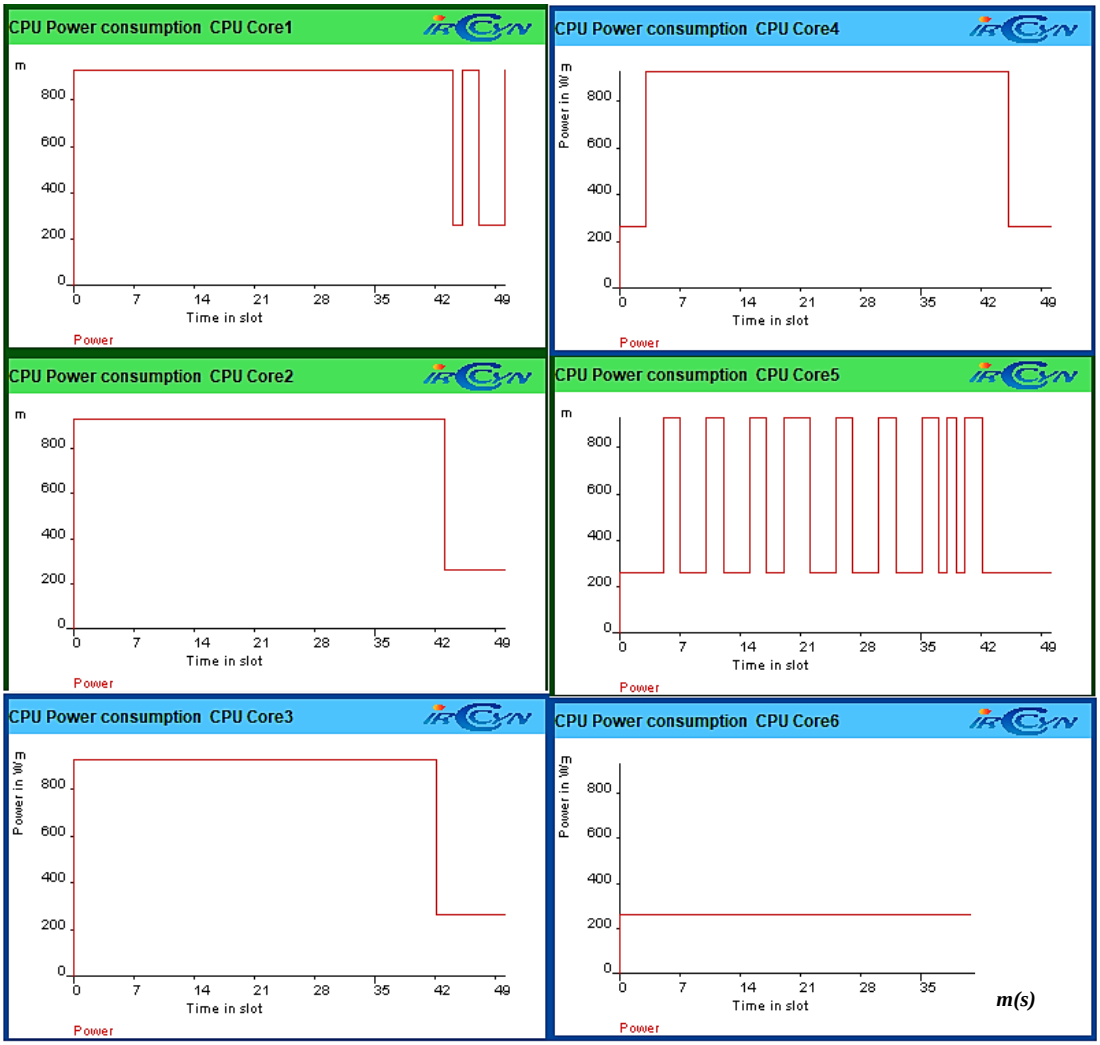


Figure 5. 47 *Power consumption of CPU using H.264 multimedia decoder application at 624 MHz*

Figure 5.48 shows the CPU load of using the H.264 multimedia decoder application at 624 MHz that illustrates the scheduling of tasks on various cores. The higher peak of the curve shows that multiple tasks $t_i \in \tau$ requires the CPU time to schedule without missing their d_i increases the utilization u_i. The fall of the curve shows that DPM

enhances the idle duration to maintain a low power mode that results in lower energy consumption using EA-EDF.

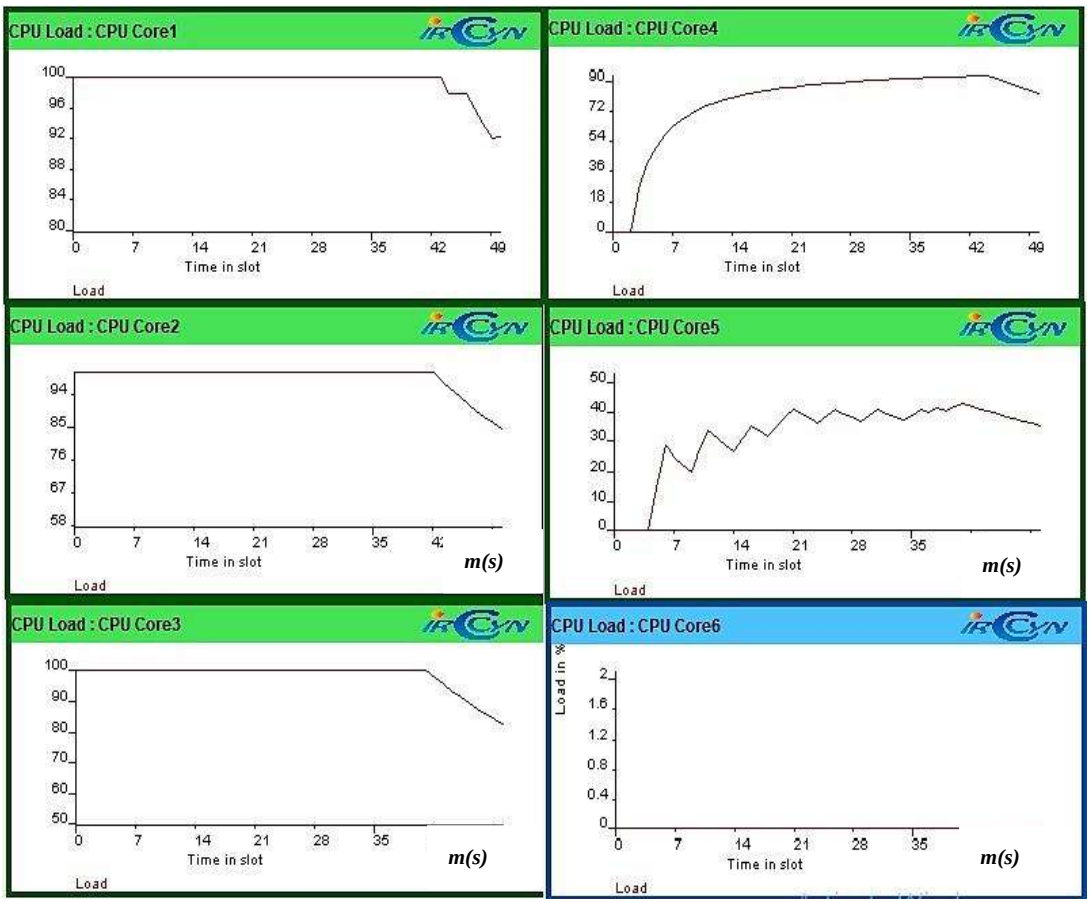


Figure 5. 48 *CPU load using H.264 multimedia decoder application at 624 MHz*

5.5 Performance Analysis of EA-EDF at 624MHz, 520MHz, 416MHz

To evaluate the performance and improvements of EA-EDF on various operating frequencies in comparison with other conventional techniques are evaluated using STORM is discussed in this section.

In Figure 5.49 the analysis compares the performance of our proposed approach EA-EDF using Intel PXA-270 MPSoC in terms of total dynamic energy dissipation for a different ready task set at various u_i utilization factors with other conventional techniques G-EDF, PDTM U-RT-DPM, TBP at 624MHz. 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 is representing the improvement % due to efficient task migration and DPM-enabled policy improving the performance by reducing energy dissipation. 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 max improvement%. The average and the min improvement% of our proposed approach compared to the GEDF approach are 4.3%, 2.25% and 0.48%, respectively. Proposed 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 execution making the dynamic power management technique more effective that's why we can see a significant improvement at (u_i=38%) 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, TBP utilization.

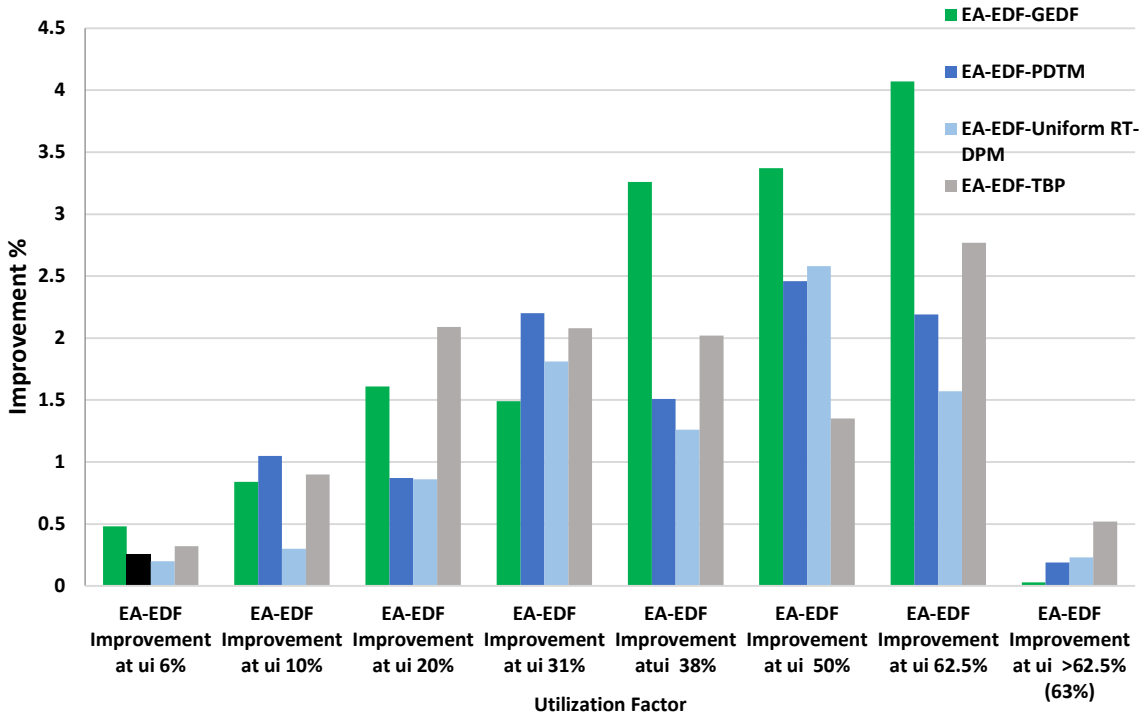


Figure 5. 49 *Performance Comparison with Conventional methods at 624MHZ*

In Figure 5.50 the analysis compares the performance of our proposed approach EA-EDF using Intel PXA-270 MPSoC in terms of total dynamic energy dissipation for the different ready task sets at various u_i utilization factors with other conventional techniques G-EDF, PDTM U-RT-DPM, TBP at 520 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 is representing the improvement % due to efficient task migration and DPM-enabled policy improving the performance by reducing energy dissipation. 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 max improvement%. The average and the min improvement% of our proposed approach compared to the GEDF approach are 4.71%, 3.81% and 0.09%, respectively.

Proposed 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 the higher number of task execution making the dynamic power management technique more effective that why we can see a significant improvement at (u_i=38%) the overall improvement is 3.67%, 2.15%, 4.7%, and 4.08% on (u_i=50%) 3.34%, 3.55%, 4.7% and 3.83% in comparison with G-EDF, PDTM U-RT-DPM, TBP utilization.

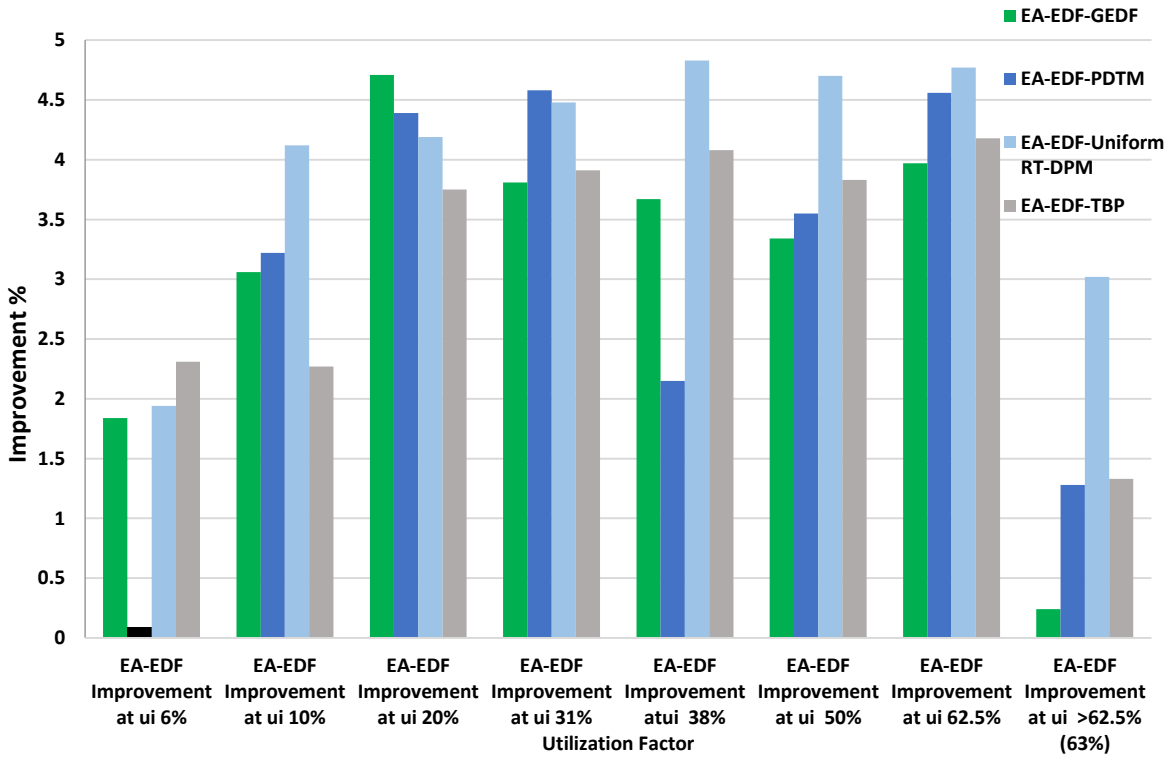


Figure 5. 50 *Performance Comparison with Conventional methods at 520MHz*

In Figure 5.51 the analysis compares the performance of our proposed approach EA-EDF using Intel PXA-270 MPSoC in terms of total dynamic energy dissipation for the different ready task sets at various u_i utilization factors with other conventional techniques G-EDF, PDTM U-RT-DPM, TBP at 416MHz. 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 is representing the improvement % due to efficient task migration and DPM-enabled policy improving the performance by reducing energy dissipation. 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 max improvement%.

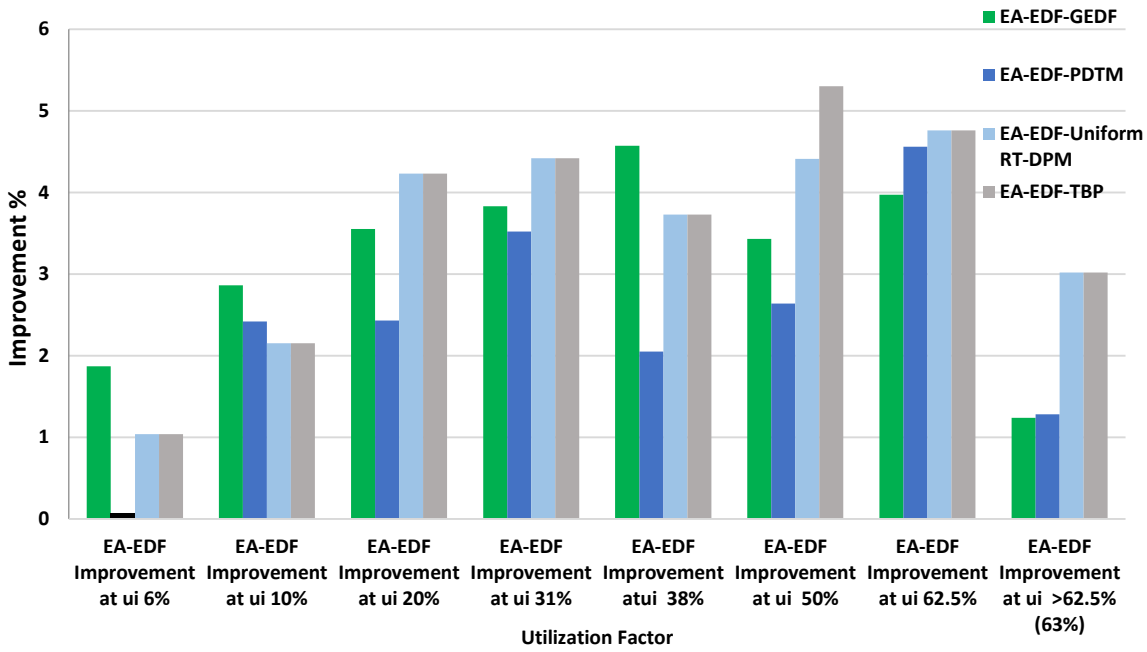


Figure 5. 51 *Performance Comparison with Conventional methods at 416MHZ*

The average and the min improvement% of our proposed approach compared to the GEDF approach are 5.3%, 4.42% and 1.04%, respectively. Proposed EA-EDF gives max improvement at (u_i=50%) on m=6 running processor, while on average improvement occurs at (u_i=38%) on m=5 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 the higher number of task execution makes the dynamic power management technique more

effective that why we can see a significant improvement at (u_i=50%) the overall improvement is 3.43%, 2.64%, 4.41% and 5.3% on (u_i=38%) 4.57%, 2.05%, 3.73% and 3.73% in comparison with G-EDF, PDTM U-RT-DPM, TBP utilization.

5.6 Summary

In this chapter, we have addressed efficient task scheduling and reduction of energy consumption using simulation of the proposed EA-EDF scheme section. The first section elaborates the complete experimental setup and its metrics. Various simulations are conducted using the proposed system model and its timing requirements based on the utilization factor are evaluated and illustrated a variety of comparative results the proposed technique gives more efficient results by optimizing 4.3, 4.7, 5.3, 5.9 and 6.3% reduction energy on various utilization factor u_i*= 6%, 10%, 20%, 31%, 45%, 55%, 62.5% and* u_i *> 62.5% (63%)* at various operating frequencies 624MHz, 520MHz, 416 MHz, 312 MHz and 208 MHz and compared to previously deployed energy optimization techniques for MPSoCs. The energy for individual core configurations is calculated and then the overall performance is measured the results depict that the proposed technique gives more efficient results. The comparison of our proposed EA-EDF with previous existing solutions allowed us to add our proposed contribution.

The proposed algorithm and the system model is evaluated and implemented in STORM, by adding the power consumption on various operating frequency of all modes of the Intel PXA270 MPSoC and its states transitions from sleep to idle and idle to running. Moreover, to take advantage of EA-EDF proposed model in this thesis developed various simulations based on the HW and SW parameters with particular variations in the no of tasks τ, the utilization factor u_i, worst case execution c_i time ratio to evaluate the reliability of the proposed algorithms. The proposed scheduler efficiently schedules tasks τ and shifts migratable tasks to the least used core to dynamically minimize available energy. Experimental evaluation shows that our technique significantly outperforms the two state-of-the-art approaches the GEDF approach and the U-RT DPM approach.

CHAPTER 6

CONCLUSIONS AND FUTURE RECOMMENDATIONS

CHAPTER 6

CONCLUSIONS AND FUTURE RECOMMENDATIONS

The final chapter summarizes the contributions made in this thesis and a descriptive overview of the whole research work and the adopted mechanism to propose an energy-aware scheduling technique using the task migration approach to reduce the energy and increase the performance of the system moreover discuss various open problems for future work.

6.1 Conclusion

The consumption of energy is one of the critical multiprocessor design considerations for multicore embedded systems. Energy-aware system increases the overall lifespan of the system, particularly for systems handling multithreaded applications. Moreover, Energy-aware scheduling schemes reduce the operating cost and also reduce thermal heat dissipation, thereby increasing the reliability of the system. Dynamic power management techniques (DPM) based on efficient scheduling with task migration (DPM) is one of the efficient mechanisms used for reducing energy consumption. Now a day's Modern embedded processors i.e MARVEL INTEL PXA-270 and PXA-250, as well as the communication subsystems, are equipped with DPM. DPM mechanism aims to reduce the consumption of energy by switching the CPU state according to the demands using the energy-power model for enabling the DPM approach. Real-time (R-T) embedded applications impose various constraints like deadline, period, worst case execution, best case execution, and priority. Embedded application observes severe consequences when the deadlines of a task are not met. In this thesis, we investigated the critical problems of improper task scheduling for application tasks with precedence and deadline constraints and higher energy dissipation.

In the first part of the thesis, various scheduling-based energy and power management strategies are discussed and critically reviewed and analyze the scheduling and computational abilities of multiprocessor systems on chip. Explored various energy-aware scheduling approaches and mapping of applications to reduce the high utilization on chip because currently high utilization u_i causes high energy consumption that causes delay and various performance degradation and high on-chip temperature issues.

In the 2nd part proposed a framework for task migration and various stages of core configurations for intelligent core switching and task migration are introduced (λ_n) is based on (u_i) to meet the deadline of the task. Moreover, we have addressed tasks that have no such restrictions to migrate to the core with less utilization. We have developed an accurate energy optimization-based task migration policy that will be used to calculate the load of the destination core. The selection of cores is based on various configurations (λ_n) have been proposed in terms of energy efficiency and utilization factor that enables a processor to reduce power and energy consumption by adopting a suitable task migration policy.

The 3rd part proposed an efficient energy-aware low-power scheduling technique based on DPM for multiprocessors that helps to attain lower energy consumption and reduces the peak utilization on MPSoC by meeting the implicit deadlines of periodic task sets $t_i \in \tau$. We have used a multiprocessor platform that illustrates the importance of dealing with a variation of utilization guarantees the schedule of τ within the time for real-time systems. Due to advancements in semiconductor technology the power densities of multiprocessors are increasing with the increase in demand for the embedded system based on complex circuitry. The exponential increase in the utilization of a multiprocessor increases the power density also increases the dissipation of energy and affects the reliability and performance of the chip. The proposed system model improves performance by reducing energy consumption by introducing low-power DPM that can efficiently increase the idle duration and optimize power to improve the reliability of a chip. The proposed algorithm schedules more tasks without missing any deadline by considering the hardware architecture of Intel PXA-270 MPSoC in the STORM simulator to control and manage the high u_i effects on the chip and also manage the energy consumption of MPSoC. To resolve these issues simulation tool for real-time multiprocessor is used to validate the energy-aware scheduling algorithm and perform scheduling by considering various cores configuration proposed in chapter 4.

The above-mentioned case is the main summary of the proposed system. The proposed system has been evaluated using STORM to minimize overall energy consumption. The chapter has presented the proposed energy model and equations to design an energy-aware DPM-based task scheduling technique using migration for achieving high-performance using EA-EDF. Chapter 05 contains the results. An increase in power utilization reduces the life span of the chip and has become an integral chip design issue

for the battery-operated multiprocessor system. The objective is to increase the performance of the multi-core systems by gradually decreasing the power and energy consumption. We have proposed the EA-EDF algorithm that enables load balancing task migration and allows processes to run on various cores at various times as decided by the scheduler using periodic task sets with implicit deadlines on multiprocessor platforms.

6.2 Future Recommendations

Although a lot of developments are made in the area of energy-aware task scheduling based on dynamic power management (DPM) and dynamic voltage and frequency scaling (DVFS) but scope of the study can be enhanced by filling the gap of several open issues in MPSoCs that needs to be considered and resolved in our future research. Reduce the total consumption of energy for multi-rate A-periodic, sporadic dependent tasks for NoC-based heterogeneous MPSoC using the task-level pipelining with DPM. Proper online task scheduling for NoC-based MPSoCs will be consider in our future research on the current problem of reducing the dynamic energy consumption for dependent tasks with deadline constraints for NoC-based combined homogeneous and heterogeneous MPSoCs that include energy-efficient online scheduling. Another industrial gap associatd to the current research is to develop an efficient task scheduling model for heterogeneous MPSoC using voltage frequency islands (VFI) based on restricted task migration. Moreover, to integrate dynamic power management (DPM) with dynamic voltage and frequency scaling (DVFS) to develop a hybrid model that can be used for m=16 core GPU-based multiprocessor platforms. Apart from this the proposed technique gives more efficient results by optimizing 4.3, 4.7, 5.3, 5.9 and 6.3% reduction energy on various u_i= 6%, 10%, 20%, 31%, 45%, 55%, 62.5% and u_i > 62.5% at 624MHz, 520MHz, 416 MHz, 312 MHz and 208 MHz operating frequency but a combined DPM-DVFS-based hybrid multiprocessing approach can be used for scheduling-based energy optimization techniques. However, the key design challenge is how optimally DPM and DVFS integrate to minimize the consumption of energy.

REFERENCES

[1] Raveendran, Arun Prasath, Jafar A. Alzubi, Ramesh Sekaran, and Manikandan Ramachandran, "A high performance scalable fuzzy based modified Asymmetric Heterogene Multiprocessor System on Chip (AHt-MPSOC) reconfigurable architecture," *Journal of Intelligent & Fuzzy Systems,* vol. 42, no. 2, pp. 647-658, 2022.

[2] Mohammadi, Mahsa, and Hakem Beitollahi, "Q-scheduler: A temperature and energy-aware deep Q-learning technique to schedule tasks in real-time multiprocessor embedded systems," *IET Computers & Digital Techniques,* vol. 2, no. 4, pp. 311-322, 2022.

[3] Dey, Somdip, et al. "CPU-GPU-Memory DVFS for Power-Efficient MPSoC in Mobile Cyber Physical Systems." *Future Internet,* vol.14. no.3, pp. 21-38, 2022.

[4] Schranzhofer, A., Chen, J. J., & Thiele, L. "Dynamic power-aware mapping of applications onto heterogeneous mpsoc platforms," *IEEE Transactions on Industrial Informatics,* vol. 6, no. 4, pp. 692-707,2010.

[5] Xie, Yi, Huihui Li, Wei Li, Yangjun Zhang, Michael Fowler, Manh Kien Tran, Xiong Zhang, Bin Chen, and Shasha Deng. "Improving thermal performance of battery at high current rate by using embedded heat pipe system," *Journal of Energy Storage,* vol.46, no.1, 2022.

[6] Nikolic, Goran, Bojan Dimitrijević, Tatjana Nikolić, and Mile Stojčev. "Fifty years of microprocessor evolution from single CPU to multicore and manycore systems," *Electronics and Energetics,* vol. 35, no. 2, pp.155-186, 2022.

[7] Aerabi, Ehsan, Mahdi Fazeli, and David Hély, "CyEnSe: Cyclic energy-aware scheduling for energy-harvested embedded systems," *Microprocessors and Microsystems,* vol. 32, no. 13, pp.122-134, 2022.

[8] Tajary, A. R, and Hossein Morshedlou. "A Simulated Annealing-based Throughput-aware Task Mapping Algorithm for Manycore Processors," *Journal of AI and Data Mining,* vol. 12, no. 2, pp.131-144, 2022.

[9] Rossi, Daniele, and Vasileios Tenentes. "Run-Time Thermal Management for Lifetime Optimization in Low-Power Designs," *Electronics,* vol.11, no. 3, pp. 411-417, 2022.

[10] Alonso, S., Lazaro, J., Jimenez, J., Muguira, L., & Bidarte, U, "Evaluating the OpenAMP framework in real-time embedded SoC platforms" *Conference on Design of Circuits and Integrated Systems (DCIS)*, pp. 1-6, 2021.

[11] H. Breit, S. Mandapati and U. Balss, "An FPGA/MPSoC Based Low Latency Onboard SAR Processor,"*IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 5159-5162, 2021.

[12] Kim, Young Geun, Seon Young Kim, Seung Hun Choi, and Sung Woo Chung. "Thermal-aware adaptive VM allocation considering server locations in heterogeneous data centers," *Journal of Systems Architecture,* vol.1. no.17,pp.111-119, 2021.

[13] D. Rupanetti and H. Salamy, "Energy Efficient Scheduling with Task Migration on MPSoC Architectures," *IEEE International Conference on Electro Information Technology (*EIT), pp. 156-161, 2019.

[14] T. Yu, R. Zhong, V. Janjic, P. Petoumenos and J. Zhai, "Collaborative heterogeneity-aware os scheduler for asymmetric multicore processors," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1224-1237, 2020.

[15] Wang, Qiang, Xinxin Mei, Hai Liu, Yiu-Wing Leung, Zongpeng Li, and Xiaowen Chu. "Energy-Aware Non-Preemptive Task Scheduling With Deadline Constraint in DVFS-Enabled Heterogeneous Clusters." *IEEE Transactions on Parallel and Distributed Systems,* vol. 72, no. 2, pp. 1224-1237, 2022.

[16] Li. Chenyun, X. Li, C. Tang and Q. Wu, "A power management circuit compatible with Ti smart reflex and xilinx ultrascale MPSoC dual voltage mechanism," *International Conference on Computing and Pattern Recognition, Shanghai, China*, pp. 357-362, 2021.

[17] Hameed, Fazal, and Jeronimo Castrillon. "BlendCache: An Energy and Area Efficient Racetrack Last-Level-Cache Architecture," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 61, no. 5, pp. 1224-1237, 2022.

[18] Young Geun, Seon Young Kim, Seung Hun Choi, and Sung Woo Chung. "Thermal-aware adaptive VM allocation considering server locations in heterogeneous data centers," *Journal of Systems Architecture,* vol.1. no.17,pp.111-119, 2021.

[19] Liu, Xun. "Low Energy Consumption and Time Deterministic Energy-saving Workflow Task Scheduling Algorithm based on DVFS," *IEEE 9th International Conference on Cyber Security and Cloud Computing (*CSCloud), pp. 56-61, 2022.

[20] Daniele, and Vasileios Tenentes. "Run-Time Thermal Management for Lifetime Optimization in Low-Power Designs," *Electronics*, vol.11, no. 3, pp. 411-417, 2022.

[21] E. Jiang, L. Wang and J. Wang, "Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 646-663, 2021.

[22] H. Wang, X. Guo, S. X. D. Tan, C. Zhang and H. Tang, "Leakage-aware predictive thermal management for multicore systems using echo state network," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 7, pp. 1400-1413, 2019.

[23] S. Z. Sheikh and M. A. Pasha, "A dynamic cache-partition schedulability analysis for partitioned scheduling on multicore real-time systems,"*IEEE Letters of the Computer Society*, vol. 3, no. 2, pp. 46-49, 2020.

[24] Park, Jurn-Gyu, Nikil Dutt, and Sung-Soo Lim. "An Interpretable Machine Learning Model Enhanced Integrated CPU-GPU DVFS Governor,"*ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 6, pp. 1-28, 2021.

[25] M. Shan and O. Khan, "Accelerating concurrent priority scheduling using adaptive in-hardware task distribution in multicores,"*IEEE Computer Architecture Letters*, vol. 20, no. 1, pp. 17-21, 2020.

[26] "1965 – "Moore's Law" Predicts the Future of Integrated Circuits". *Computer History*.

[27] S. E. Abdellaoui and Y. Fakhri, "Power management strategies in energy-harvesting wireless sensor networks," *International Journal of Communication Networks and Information Security*, vol. 13, no. 1, pp. 136-142, 2021.

[28] Sandhu, Muhammad Moid, Sara Khalifa, Raja Jurdak, and Marius Portmann, "Task Scheduling for Energy-Harvesting-Based IoT: A Survey and Critical Analysis,"*IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13825-13848, 2021.

[29] El Yaacoub, Ahmed, Luca Mottola, Thiemo Voigt, and Philipp Rümmer. "Scheduling Dynamic Software Updates in Safety-critical Embedded Systems-the Case of Aerial Drones," *ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*, pp. 284-285, 2022.

[30] L. E. Rubio-Anguiano, A. C. Trabanco, J. L. B. Velasco and A. Ramírez-Treviño, "Maximizing Utilization and Minimizing Migration in Thermal-Aware Energy-Efficient Real-Time Multiprocessor Scheduling,"*IEEE Access*, vol. 9, pp. 83309-83328, 2021.

[31] Radecki, Andrzej, and Tomasz Rybicki. "Simulation Oriented Layer of Embedded Software Architecture for Rapid Development of Custom Embedded Systems Virtual Simulators Used in Didactics," *Applied Sciences*, vol. 12, no. 13, pp. 111-127, 2022.

[32] Roeder, Julius, Benjamin Rouxel, Sebastian Altmeyer, and Clemens Grelck, "Energy-aware scheduling of multi-version tasks on heterogeneous real-time systems,"*In Proceedings of the 36th Annual ACM Symposium on Applied Computing*, pp. 501-510, 2021.

[33] Salim, A., Mohammed Ali, and Emad H. Al-Hemairy, "Communication Module for V2X Applications using Embedded Systems," *In Journal of Physics*, vol. 19, no. 1,pp . 115-123 2021.

[34] P. Fruehauf, A. Munding, K. Pressel, P. Schwarz and M. Vogt, "Chip-package-board reliability of system-in-package using laminate chip embedding technology based on lead frame,"*IEEE Transactions on Components Packaging and Manufacturing Technology*, vol. 10, no. 1, pp. 44-56, 2019.

[35] Qureshi, Kashif Naseer, Ejaz Ahmad, Muhammad Anwar, Kayhan Zrar Ghafoor, and Gwanggil Jeon, "Network functions virtualization for mobile core and heterogeneous cellular networks," *Wireless Personal Communications*, vol.122, no. 3, PP.2543-2559, 2022.

[36] W. Yankai, S. Wang, B. Yang, L. Zhu and F. Liu, "Big data driven hierarchical digital twin predictive remanufacturing paradigm: architecture, control mechanism, application scenario and benefits," *Journal of Cleaner Production*, vol. 248, no. 1, pp. 119-299, 2020.

[37] Liu, Chunhui, Kai Liu, Hualing Ren, Xincao Xu, Ruitao Xie, and Jingjing Cao," real-time distributed strategy for multi-period task offloading in vehicular edge computing environment," *Neural Computing and Applications*, vol. 15, no 3, PP. 1-15, 2021.

[38] Yuda, Irawan, Wahyuni Refni, Muhardi Muhardi, Fonda Hendry, Muzawi Rometdo, and Luthfi Hamzah Muhammad,"Real time system monitoring and analysis-based internet of things (IoT) technology in measuring outdoor air quality," *International Journal of Interactive Mobile Technologies*, vol. 15, no. 10, pp.220-235, 2021.

[39] Z. Xiaorui, W. Zhang, W. Sun, H. Wu, A. Song et al., "A real-time cutting model based on finite element and order reduction,"*Computer Systems Science and Engineering*, vol. 43, no.1, pp.1-15, 2022.

[40] Andrzej, and Tomasz Rybicki. "Simulation Oriented Layer of Embedded Software Architecture for Rapid Development of Custom Embedded Systems Virtual Simulators Used in Didactics," *Applied Sciences*, vol. 12, no. 13, pp. 111-127, 2022.

[41] Ruparelia, Stavan, Monil Jethva, and Ruchi Gajjar,"Real-Time Tomato Detection, Classification, and Counting System Using Deep Learning and Embedded Systems," *In Proceedings of the International e-Conference on Intelligent Systems and Signal Processing*, pp. 511-522, 2022.

[42] Julius, Roeder, Benjamin Rouxel, Sebastian Altmeyer, and Clemens Grelck, "Energy-aware scheduling of multi-version tasks on heterogeneous real-time systems,"*In Proceedings of the 36th Annual ACM Symposium on Applied Computing*, pp. 501-510, 2021.

[43] Beamonte, Raphael, Naser Ezzati-Jivan, and Michel R. Dagenais, "Execution trace-based model verification to analyze multicore and real-time systems," *Concurrency and Computation: Practice and Experience*, vol. 4, no. 8, pp. 220-235, 2022.

[44] Wang, Jinwen, Ao Li, Haoran Li, Chenyang Lu, and Ning Zhang,"RT-TEE: Real-time System Availability for Cyber-physical Systems using ARM TrustZone" *In 2022 IEEE Symposium on Security and Privacy (SP), IEEE Computer Society*, pp. 1573-1573, 2022.

[45] Salman, Shaik Mohammed, Alessandro V. Papadopoulos, Saad Mubeen, and Thomas Nolte. "Multi-processor scheduling of elastic applications in compositional real-time systems." *Journal of Systems Architecture*, vol. 22, no. 10, pp.210-223, 2022

[46] Hanumantha Rao, and Naresh kumar reddy Beechu. "An Efficient Real-Time Embedded Application Mapping For NoC Based Multiprocessor System on Chip" vol. 22, no. 10, pp.210-223, 2022.

[47] Chen, Yang, Lin Liu, Xuelin Feng, and Jinglin Shi, "DXT501: An SDR-Based Baseband MP-SoC for Multi-Protocol Industrial Wireless Communication," *In 2022 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS)*, pp. 1-6, 2022.

[48] He, J., Xiao, Y., Bogdan, C., Nazarian, S., & Bogdan, P. "A Design Methodology for Energy-Aware Processing in Unmanned Aerial Vehicles," *ACM Transactions on Design Automation of Electronic Systems*, pp. 1-20 vol. 27, 2021,.

[49] Branco, K. R., Pelizzoni, J. M., Neris, L. O., Trindade, O., Osório, F. S., & Wolf, D. F. Tiriba, "A new approach of uav based on model driven development and

multiprocessors *IEEE International Conference on Robotics and Automation"*, pp. 1-4,2011.

[50] Coskun, A. K., Rosing, T. S., Mihic, K., De Micheli. Y,"Analysis and optimization of MPSoC reliability," *Journal of Low Power Electronics,* , vol. 2(1), 56-69.

[51] Son, Hyun-Sik, Deok-Keun Kim, Seung-Hwan Yang, and Young-Kiu Choi,"Real-time power line detection for safe flight of agricultural spraying drones using embedded systems and deep learning,"*IEEE Access*, vol. 144, no 2. pp.287-304,2022.

[52] Liu, Xiaojun. "Design of domestic embedded computer system based on ARM." In 2021 4th International Conference on Information Systems and Computer Aided Education, pp. 2435-2439, 2021.

[53] Ali, Haider, Umair Ullah Tariq, James Hardy, Xiaojun Zhai, Liu Lu, Yongjun Zheng, Faycal Bensaali, Abbes Amira, Kaniz Fatema, and Nikos Antonopoulos,"A survey on system level energy optimisation for MPSoCs in IoT and consumer electronics,"*Computer Science Review*, vol. 41, no 2, pp. 225-243 2021.

[54] Tang, Kunhao, Wei Jiang, Ruonan Cui, and Youlong Wu,"A memory-based task scheduling algorithm for grid computing based on heterogeneous platform and homogeneous tasks,"*International Journal of Web and Grid Services*,vol.16, no.3, pp.287-304, 2020.

[55] Tang, Qi, Li-Hua Zhu, Li Zhou, Jun Xiong, and Ji-Bo Wei, "Scheduling directed acyclic graphs with optimal duplication strategy on homogeneous multiprocessor systems,"*Journal of Parallel and Distributed Computing*,vol.138, no.2, pp. 115-127, 2020.

[56] Gourisaria, Mahendra Kumar, Pabitra Mohan Khilar, and Sudhansu Shekhar Patra, "EPTS: Energy-saving pre-emptive task scheduling for homogeneous cloud systems," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 24, no. 8, pp. 2415-2441, 2021.

[57] Zhang, J., Cheng, G., Lu, C., Guo, T., Kang, J., Yan, X., ... & Yuan, X, "Flow Data Task Scheduling Model of RTOS Based on Multicore Operation System," *IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pp. 56-60, 2021.

[58] Alonso, S., J. Lázaro, J. Jiménez, U. Bidarte, and L. Muguira, "Evaluating Latency in Multi-Processing Embedded Systems for the Smart Grid," *Energies*, vol 14, no.3, pp 3322-3340, 2021.

[59] Bertout, Antoine, "Measurement-Based Timing Analysis on Heterogeneous MPSoCs:A Practical Approach," *In Software Architecture: 14th European Conference, ECSA 2020 Tracks and Workshops, L'Aquila, Italy*, Proceedings, pp. 279-285, 2020.

[60] Dejian, Li, Yang LiXin, Bai Zhihua, He Longlong, Zhang Guang, and Li Meng. "Asymmetric hardware and software integration design based on multi-core processor,"*In 2021 International Conference on Communications, Information System and Computer Engineering (CISCE)*, pp. 109-113. 2021.

[61] Mahmood, Amjad, Salman A. Khan, Fawzi Albalooshi, and Noor Awwad,"Energy-aware real-time task scheduling in multiprocessor systems using a hybrid genetic algorithm," *Electronics*, vol.6, no. 2, pp. 67-88, 2017.

[62] Choudhury, Adil Mahmud, and Kamruddin Nur,"Qalitative Study of Contention-aware Scheduling Algorithm for Asymmetric Multicore Processors," In Proceedings of the International Conference on Computing Advancements, pp. 1-6, 2020.

[63] Jamil, Roy, Emmanuel Grolleau, Bernard Dautrevaux, and Antoine Bertout. "Measurement-based timing analysis on heterogeneous mpsoc: A practical approach,"*In European Conference on Software Architecture*, pp. 279-293, 2020.

[64] Parane, Khyamling, Prabhu Prasad, and Basavaraj Talawar, "P-NoC: Performance Evaluation and Design Space Exploration of NoCs for Chip Multiprocessor Architecture Using FPGA," *Wireless Personal Communication*, vol. 114, no. 4, pp. 3295-3319,2020.

[65] Saha, Sangeet, Shounak Chakraborty, Xiaojun Zhai, Shoaib Ehsan, and Klaus McDonald-Maier, "ACCURATE: Accuracy Maximization for Real-Time Multi-core systems with Energy Efficient Way-sharing Caches," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol.6, no. 2, pp. 67-88, 2022.

[66] Grüttner, Kim, Philipp A. Hartmann, Tiemo Fandrey, Kai Hylla, Daniel Lorenz, Stefan Hauck-Stattelmann,"A Timed-Value Stream Based ESL Timing and Power Estimation and Simulation Framework for Heterogeneous MPSoCs," *International Journal of Parallel Programming*, vol.48, no. 6, pp. 957-1007, 2020.

[67] He, Zhenli, Ying Sun, Hui Fang, Yi-Xiong Huang, Yu Yang, Zhi-Yuan Zhang, and Di Liu, "Energy-Efficient System Design of Asymmetric Multiprocessor for Real-Time Streaming Applications,"*In 2021 International Conference on Intelligent Technology and Embedded Systems (ICITES)*, pp. 44-51, 2021.

[68] Hossain, Md Shazzad, and Ioannis Savidis, "Leakage Reuse for Energy Efficient Near-Memory Computing of Heterogeneous DNN Accelerators," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 11, no. 4, pp. 762-775, 2021.

[69] Ali, Haider, Umair Ullah Tariq, James Hardy, Xiaojun Zhai, Liu Lu, Yongjun Zheng, Faycal Bensaali, Abbes Amira, Kaniz Fatema, and Nikos Antonopoulos, "A survey on system level energy optimisation for MPSoCs in IoT and consumer electronics,"*Computer Science Review*" vol. 41, no. 6, pp. 911-927, 2021.

[70] Mahmud, Umar, Shariq Hussain, and Ibrahima Kalil Toure,"Gathering Contextual Data with Power Information Using Smartphones in Internet of Everything," *Wireless Communications and Mobile Computing*, vol 12, no.4, pp. 121-137, 2022.

[71] Nambinina, Rakotojaona, Daniel Onwuchekwa, Hamidreza Ahmadian, Dinesh Goyal, and Roman Obermaisser, "Time-Triggered Frequency Scaling in Network-on-Chip for Safety-Relevant Embedded Systems," *International Conference on Smart Generation Computing, Communication and Networking (SMART* GENCON), pp. 1-7, 2021.

[72] S Kariman, A. R., M. B. Ghaznavi-Ghoushchi, and A. R. Shamshiri. "An energy-efficient threshold voltage extractor with the burst-mode operation and extended long-term variations using FVF." Microelectronics Journal vol 104, 2020.

[73] Taheri, Golnaz, Ahmad Khonsari, Reza Entezari-Maleki, and Leonel Sousa. "Temperature-aware core management in MPSoCs: modelling and evaluation using MRMs." IET Computers & Digital Techniques vol 14, no. 1, PP 17-26, 2020.

[74] Arora, D., Ravi, S., Raghunathan, A., & Jha, N. K. "Architectural Support for Run-Time Validation of Program Data Properties". IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 15, no. (5), 546-559, 2007

[75] L. E. Rubio-Anguiano, J. L. Briz and A. Ramírez-Treviño, "Accounting for Preemption and Migration Costs in the Calculation of Hard Real-Time Cyclic Executives for MPSoCs," in IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 7990-7997, July 2022.

[76] L. E. Rubio-Anguiano, A. C. Trabanco, J. L. B. Velasco and A. Ramírez-Treviño, "Maximizing Utilization and Minimizing Migration in Thermal-Aware Energy-Efficient Real-Time Multiprocessor Scheduling,"*IEEE Access*, vol. 9, no 2, pp. 83309-83328, 2021.

[77] E. Jiang, L. Wang and J. Wang, "Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks, "*Tsinghua Science and Technology*, vol. 26, no. 5, pp. 646-663, 2021.

[78] Amini Motlagh, Aida, Ali Movaghar, and Amir Masoud Rahmani, "Task scheduling mechanisms in cloud computing: A systematic review," *International Journal of Communication Systems*, vol. 33, no. 6, pp. 446-460 2020.

[79] M. Shan and O. Khan, "Accelerating concurrent priority scheduling using adaptive in-hardware task distribution in multicores,"*IEEE Computer Architecture Letters*, vol. 20, no. 1, pp. 17-21, 2020.

[80] Y. Cui, W. Zhang, V. Chaturvedi and B. He, "Decentralized Thermal-Aware Task Scheduling for Large-Scale Many-Core Systems,"*IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 6, pp. 2075-2088, 2016.

[81] Bhat, Ganapati, Suat Gumussoy, and Umit Y. Ogras,"Analysis and Control of Power–Temperature Dynamics in Heterogeneous Multiprocessors," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 1, pp. 329-341, 2020.

[82] Z. Albayati, Y. Sun, H. Zeng, M. D. Natale, Q. Zhu, and B. H. Meyer, "Partitioning and selection of data consistency mechanisms for multicore real- time systems," *ACM Trans. Embedded Comput. Syst*, vol. 18, no. 4, pp. 35-46, 2019.

[83] L. Wang, E. Jiang, and J. Wang, "Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks, "*Tsinghua Science and Technology*, vol. 26, no. 5, pp. 646-663, 2021.

[84] Singh, Amit Kumar, Somdip Dey, Klaus McDonald-Maier, and Bashir M. Al-Hashimi,"Dynamic energy and thermal management of multi-core mobile platforms: A survey." *IEEE Design & Test*, vol.37, no. 5, pp. 25-33, 2020.

[85] J. Sun, N. Guan, S. Chang, F. Li, Q. Deng and W. Yi, "Capacity Augmentation Function for Real-Time Parallel Tasks With Constrained Deadlines Under GEDF Scheduling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 4537-4548,2020.

[86] Gao, Nan, Cheng Xu, Xin Peng, Haibo Luo, Wufei Wu, and Guoqi Xie. "Energy-efficient scheduling optimization for parallel applications on heterogeneous distributed systems," *Journal of Circuits, Systems and Computers*, vol. 29, no. 13, pp. 437-448, 2020.

[87] Qiu, Weiming, Yonghao Chen, Dihu Chen, Tao Su, and Simei Yang, "Run-Time Hierarchical Management of Mapping, Per-Cluster DVFS and Per-Core DPM for Energy Optimization," *Electronics*, vol 11, no. 7, pp. 215-233, 2022.

[88] Sun, Yidan, Guiyuan Jiang, Siew-Kei Lam, and Fangxin Ning, "Designing energy-efficient MPSoC with untrustworthy 3PIP cores," *IEEE Transactions on Parallel and Distributed Systems*, vol.31, no. 1, pp. 51-63,2019.

[89] P. M. Kumar and R. Nagendra, "Simulation of real time multiprocessor static scheduling algorithms,"*i-Manager's Journal on Embedded Systems*, vol. 7, no. 1, pp. 30-38, 2018.

[90] Javaid, H. Shafique, M., Henkel, J., & Parameswaran, S, "System-level application-aware dynamic power management in adaptive pipelined MPSoCs for multimedia, "*IEEE/ACM International Conference on Computer-Aided Design*, pp. 616-623, 2011.

[91] Hosahalli, Doreswamy, and Kunal G. Srinivas,"Enhanced reinforcement learning assisted dynamic power management model for internet-of-things centric wireless sensor network,"*IET Communications*, vol. 14, no. 21, pp.3748-3760, 2020.

[92] Huang, Kai, Ke Wang, Dandan Zheng, Xiaowen Jiang, Xiaomeng Zhang, Rongjie Yan, and Xiaolang Yan, "Expected energy optimization for real-time multiprocessor socs running periodic tasks with uncertain execution time,"*IEEE Transactions on Sustainable Computing*, vol. 6, no. 3, pp. 398-4112018.

[93] Singh, Amit Kumar, Somdip Dey, Klaus McDonald-Maier, Karunakar Reddy Basireddy, Geoff V. Merrett, and Bashir M. Al-Hashimi,"Dynamic energy and thermal management of multi-core mobile platforms: A survey," *IEEE Design & Test*,vol. 37, no. 5, pp. 25-33,2020.

[94] Huang, Kai, Ke Wang, Dandan Zheng, Xiaowen Jiang, Xiaomeng Zhang, Rongjie Yan, and Xiaolang Yan, "Expected energy optimization for real-time multiprocessor socs running periodic tasks with uncertain execution time." *IEEE Transactions on Sustainable Computing*, vol. 6, no. 3, pp.398-411, 2018.

[95] Marvell PXA270 Processor Electrical, mechanical and Thermal Specification data Sheet, *http://www.marvell.com/application-processors/pxafamily/assets/pxa-27x-emts.pdf*, 2009.

[96] Y. Aslan, J. Puskely, J. H. J. Janssen, M. Geurts, A. Roederer and A. Yarovoy, "Thermal-Aware Synthesis of 5G Base Station Antenna Arrays: An Overview and a Sparsity-Based Approach," *IEEE Access*, vol. 6, pp. 58868-58882, 2018.

[97] Safari, Sepideh, Mohsen Ansari, Heba Khdr, Pourya Gohari-Nazari, and Jörg Henkel, "A Survey of Fault-Tolerance Techniques for Embedded Systems From the Perspective of Power, Energy, and Thermal Issues," *IEEE Access*, vol.10,no.3, pp. 12229-12251 2022.

[98] Huang, Kai, Ke Wang, Dandan Zheng, Xiaowen Jiang, Xiaomeng Zhang, Rongjie Yan, and Xiaolang Yan, "Expected energy optimization for real-time multiprocessor socs running periodic tasks with uncertain execution time,"*IEEE Transactions on Sustainable Computing*, vol. 6, no. 3, pp.398-411,2018.

[99] Haotian, Wu, Guo Ruifeng, Deng Changyi, Peng Azhen, and Zheng Liaomo,"A low power scheduling algorithm with balanced factor for periodic tasks." *International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pp. 1-6, 2017.

[100] Hossain, MD Shazzad, and Ioannis Savidis.,"Dynamic idle core management and leakage current reuse in MPSoC platforms," *In Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, pp. 49-54, 2020.

[101] Ali, Haider, Umair Ullah Tariq, James Hardy, Xiaojun Zhai, and Nikos Antonopoulos, "A survey on system level energy optimisation for MPSoCs in IoT and consumer electronics," *Computer Science Review*, vol.41,no.4, pp.115-130, 2021.

[102] Huang, Kai, Ke Wang, Dandan Zheng, Xiaowen Jiang, Xiaomeng Zhang, Rongjie Yan, and Xiaolang Yan. "Expected energy optimization for real-time multiprocessor socs running periodic tasks with uncertain execution time," *IEEE Transactions on Sustainable Computing*, vol. 6, no. 3, pp. 398-411, 2018.

[103] Zhang and Yi-wen, "System level fixed priority energy management algorithm for embedded real time application," *Microprocessors and Microsystems*, vol, 64, no.2 pp. 170-177, 2019.

[104] Ma, Yue, Junlong Zhou, Thidapat Chantem, and Xiaobo Sharon Hu, "Online Resource Management for Improving Reliability of Real-Time Systems on Big–Little Type MPSoCs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 1, pp. 88-100, 2018.

[105] Esmaili, Amirhossein, Mahdi Nazemi, and Massoud Pedram,"Modeling processor idle times in MPSoC platforms to enable integrated DPM, DVFS, and task scheduling subject to a hard deadline." *In Proceedings of the 24th Asia and South Pacific Design Automation Conference*, pp. 532-537, 2019.

[106] Torres, Carlos Cesar Cortes, Hayate Okuhara, Nobuyuki Yamasaki, and Hideharu Amano, "Analysis of Body Bias Control Using Overhead Conditions for Real Time Systems:A Practical Approach," *IEICE Transactions on Information and Systems*, vol. 101, no. 4, pp. 1116-1125, 2018.

[107] K. Kaur, Savina. Bansal, and Rakesh. Bansal,"Duplication-controlled static energy-efficient scheduling on multiprocessor computing system," Concurrency and Computation: *Practice and Experience*, vol. 29, no. 12, pp. 111-125 2017.

[108] Aldegheri, Stefano, Nicola Bombieri, and Hiren Patel, "On the task mapping and scheduling for DAG-based embedded vision applications on heterogeneous multi/many-core architectures," *In 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1003-1006, 2020.

[109] K. Huang et al., "A Scalable and Adaptable ILP-Based Approach for Task Mapping on MPSoC Considering Load Balance and Communication Optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 9, pp. 1744-1757, 2019.

[110] W. Liu, J. Yi, M. Li, P. Chen and L. Yang, "Energy-Efficient Application Mapping and Scheduling for Lifetime Guaranteed MPSoCs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 1, pp. 1-14,2019.

[111] B. Wang, Y. Song, C. Wang, W. Huang and X. Qin, "A Study on Heuristic Task Scheduling Optimizing Task Deadline Violations in Heterogeneous Computational Environments," *IEEE Access*, vol. 8, no.2. pp. 205635-205645, 2020.

[112] K. Huang, L. Santinelli, and G. C. Buttazzo, "Adaptive Dynamic Power Management for Hard Real-Time Systems," *30th IEEE Real-Time Systems Symposium*, pp. 23-32, 2009.

[113] K. Huang, L. Santinelli, J. -J. Chen, L. Thiele and G. C. Buttazzo, "Periodic power management schemes for real-time event streams," *Proceedings of the 48h IEEE Conference on Decision and Control (CDC)*, pp. 6224-6231,2009.

[114] A. Rowe, K. Lakshmanan, H. Zhu and R. Rajkumar,"Rate-Harmonized Scheduling and Its Applicability to Energy Management," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 3, pp. 265-275,2010,

[115] Q. Qi et al., "Scalable Parallel Task Scheduling for Autonomous Driving Using Multi-Task Deep Reinforcement Learning,"*IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13861-13874,2020.

[116] K. Baital and A. Chakrabarti,"Dynamic Scheduling of Real-Time Tasks in Heterogeneous Multicore Systems," *IEEE Embedded Systems Letters*, vol. 11, no. 1, pp. 29-32, 2019.

[117] Cheol-Hoon Lee and K. G. Shin, "On-line dynamic voltage scaling for hard real-time systems using the EDF algorithm," *25th IEEE International Real-Time Systems Symposium*, pp. 319-335, 2004.

[118] M. A. Awan and S. M. Petters, "Enhanced Race-To-Halt: A Leakage-Aware Energy Management Approach for Dynamic Priority Systems,"*23rd Euromicro Conference on Real-Time Systems*, pp. 92-101. 2011.

[119] A. Schranzhofer, J. Chen and L. Thiele, "Dynamic Power-Aware Mapping of Applications onto Heterogeneous MPSoC Platforms," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 692-707, 2010.

[120] X. Jiang, J. Sun, Y. Tang and N. Guan, "Utilization-Tensity Bound for Real-Time DAG Tasks under Global EDF Scheduling,"*IEEE Transactions on Computers*, vol. 69, no. 1, pp. 39-50,2020.

[121] L. E. Rubio-Anguiano, A. C. Trabanco, J. L. B. Velasco and A. Ramírez-Treviño, "Maximizing Utilization and Minimizing Migration in Thermal-Aware Energy-Efficient Real-Time Multiprocessor Scheduling,"*IEEE Access*, vol. 9,no.1. pp. 83309-83328, 2021.

[122] H. Ali, U. U. Tariq, Y. Zheng, X. Zhai and L. Liu, "Contention & Energy-Aware Real-Time Task Mapping on NoC Based Heterogeneous MPSoCs," *IEEE Access*, vol. 6, no.2. pp. 75110-75123, 2018.

[123] J. -J. Han, X. Tao, D. Zhu, H. Aydin, Z. Shao and L. T. Yang, "Multicore Mixed-Criticality Systems: Partitioned Scheduling and Utilization Bound," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 1, pp. 21-34, 2018.

[124] Bansal, Savina, Rakesh Kumar Bansal, and Kiran Arora, "Energy-cognizant scheduling for preference-oriented fixed-priority real-time tasks," *Journal of Systems Architecture*, vol.108, no 3, pp. 21-34, 2020.

[125] Chang, Naehyuck, Enrico Macii, Massimo Poncino, and Vivek Tiwari,"System-Level Power Management," *EDA for IC System Design, Verification, and Testing*, pp, 7-1,2018.

[126] Kumari, Raj, Sakshi Kaushal, and Naveen Chilamkurti, "Energy conscious multi-site computation offloading for mobile cloud computing." Soft Computing 22, no. 2, pp.6751-6764, 2018.

[127] Wang, Yanting, Min Sheng, Xijun Wang, and Jiandong Li, "Cooperative dynamic voltage scaling and radio resource allocation for energy-efficient multiuser mobile edge computing," *IEEE International Conference on Communications (ICC)*, pp. 1-6, 2018.

[128] Yang J, Zhou X, Chrobak M, Zhang Y, Jin L, "Dynamic thermal management through task scheduling. Performance analysis of systems and software," *IEEE international symposium*, pp. 191–201. 2008.

[129] P. Wang et al., "Dynamic Thermal Analysis of High-Voltage Power Cable Insulation for Cable Dynamic Thermal Rating," *IEEE Access*, vol. 7, no.4. pp. 56095-56106, 2019.

[130] C. Qian et al., "Thermal Management on IGBT Power Electronic Devices and Modules," *IEEE Access*, vol. 6, no.1. pp. 12868-12884, 2018.

[131] Y. Fu, L. Li, K. Wang and C. Zhang, "Kalman Predictor-Based Proactive Dynamic Thermal Management for 3-D NoC Systems With Noisy Thermal Sensors,"*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 11, pp. 1869-1882, 2017.

[132] S. Rahimipour, W. N. Flayyih, N. A. Kamsani, S. J. Hashim, M. R. Stan and F. Z. B. Rokhani,"Low-Power, Highly Reliable Dynamic Thermal Management by Exploiting Approximate Computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 10, pp. 2210-2222, 2020.

[133] R. M. Vignali, F. Borghesan, L. Piroddi, M. Strelec and M. Prandini, "Energy Management of a Building Cooling System With Thermal Storage: An Approximate Dynamic Programming Solution," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 619-633, 2017.

[134] S. Maity, A. Ghose, S. Dey and S. Biswas, "Thermal Load-aware Adaptive Scheduling for Heterogeneous Platforms,"*19th International Conference on Embedded Systems (VLSID)*, pp. 125-130, 2020.

[135] M. A. Akram, I. -C. Hwang and S. Ha, "Power Delivery Networks for Embedded Mobile SoCs: Architectural Advancements and Design Challenges," *IEEE Access*, vol. 9, pp. 46573-46588, 2021.

[136] Kabir, Rashadul, and Baback Izadi, "Naïve Bayesian based Temperature and Energy Aware Scheduling of Heterogeneous Processors," *Tenth International Green and Sustainable Computing Conference (IGSC)*, pp. 1-8, 2019.

[137] Mao, Jianzhou, Tathagata Bhattacharya, Xiaopu Peng, Ting Cao, and Xiao Qin. "Modeling energy consumption of virtual machines in dvfs-enabled cloud data centers," *IEEE 39th International Performance Computing and Communications Conference (IPCCC)*, pp. 1-6, 2020.

[138] M. Asif et al., "A High Performance Adaptive Digital LDO Regulator With Dithering and Dynamic Frequency Scaling for IoT Applications," *IEEE Access*, vol. 8, pp. 132200-132211, 2020.

[139] Nunez-Yanez, Jose. "Energy Proportional Heterogenous Computing with Reconfigurable MPSoC." *In 2019 International Conference on High Performance Computing & Simulation (HPCS)*, pp. 642-642, 2019.

[140] Y. Lu and W. He, "A Dual-rail Based Dynamic Voltage and Frequency Scaling for Wide-Voltage-Range Processor," *2021 IEEE 14th International Conference on ASIC (ASICON)*, pp. 1-4, 2021.

[141] M. A. Akram, I. -C. Hwang and S. Ha, "Power Delivery Networks for Embedded Mobile SoCs: Architectural Advancements and Design Challenges," *IEEE Access*, vol. 9, pp. 46573-46588, 2021.

[142] Ansari, M., Yeganeh-Khaksar, A., Safari, S. and Ejlali, "Peak-power-aware energy management for periodic real-time applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 4, pp.779-788, 2019.

[143] Yifan Zhu and Frank Mueller,"Feedback EDF scheduling of real-time tasks exploiting dynamic voltage scaling," *Journal on Real-Time Systems*, vol 5,no 1,pp. 112-123, 2005.

[144] Yifan Zhu and F. Mueller, "Feedback EDF scheduling exploiting dynamic voltage scaling," *Proceedings. RTAS IEEE Real-Time and Embedded Technology and Applications Symposium*, pp. 84-93, 2004.

[145] M. Bambagini, F. Prosperi, M. Marinoni and G. Buttazzo, "Energy management for tiny real-time kernels," *International Conference on Energy Aware Computing*, pp. 1-6, 2011.

[146] Taheri, Golnaz, Ahmak Khonsari, Reza Entezari-Maleki, Mohammad Baharloo, and Leonel Sousa, "Temperature-aware dynamic voltage and frequency scaling

enabled MPSoC modeling using stochastic activity networks," *Microprocessors and Microsystems*,vol.60, no 2, pp. 15-23, 2018.

[147] Iranfar, Arman, Mehdi Kamal, Ali Afzali-Kusha, Massoud Pedram, and David Atienza, "Thespot: Thermal stress-aware power and temperature management for multiprocessor systems-on-chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 8, pp. 1532-1545, 2017.

[148] Iskandar, Veronia, Cherif Salama, and Mohamed Taher, "Dynamic thread mapping for maximizing performance in power-efficient multi-core systems," *13th International Conference on Computer Engineering and Systems (*ICCES), pp. 230-235, 2018.

[149] Ma, Yue, Junlong Zhou, Thidapat Chantem, Robert P. Dick, Shige Wang, and Xiaobo Sharon Hu. "Online Resource Management for Improving Reliability of Real-Time Systems on "Big–Little" Type MPSoCs,"*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 1, pp. 88-100, 2018.

[150] Z. Ren, J. C. Kim and J. Lee, "Transient Cooling and Heating Effects in Holey Silicon-Based Lateral Thermoelectric Devices for Hot Spot Thermal Management," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 11, no. 8, pp. 1214-1222, 2021.

[151] P. Wang et al., "Dynamic Thermal Analysis of High-Voltage Power Cable Insulation for Cable Dynamic Thermal Rating," *IEEE Access*, vol. 7, pp. 56095-56106, 2019.

[152] L. Cheng, K. Huang, G. Chen, Z. Bing and A. C. Knoll, "Adaptive Periodic Thermal Management for Pipelined Hard Real-Time Systems," *IEEE Access*, vol. 7, pp. 114731-114746, 2019.

[153] H. Wang, L. Hu, X. Guo, Y. Nie and H. Tang, "Compact Piecewise Linear Model Based Temperature Control of Multicore Systems Considering Leakage Power," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 12, pp. 7556-7565, 2020.

[154] C. Yang et al., "MLD-Based Thermal Behavior Analysis of Traction Converters Under Faulty Conditions," *IEEE Transactions on Transportation Electrification*, vol. 7, no. 3, pp. 1058-1073, Sept. 2021.

[155] K. -C. Chen, H. -W. Tang, Y. -H. Liao and Y. -C. Yang, "Temperature Tracking and Management With Number-Limited Thermal Sensors for Thermal-Aware NoC Systems," *IEEE Sensors Journal*, vol. 20, no. 21, pp. 13018-13028, 2020.

[156] S. A. A. Bukhari, F. Khalid, O. Hasan, M. Shafique and J. Henkel, "Toward Model Checking-Driven Fair Comparison of Dynamic Thermal Management Techniques Under Multithreaded Workloads,"*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 8, pp. 1725-1738,2020.

[157] R. Cheng et al., "Stochastic Dynamic Programming-Based Online Algorithm for Energy Management of Integrated Energy Buildings With Electric Vehicles and Flexible Thermal Loads," *IEEE Access*, vol. 9, pp. 58780-58789, 2021.

[158] Y. G. Kim, M. Kim, J. Kong and S. W. Chung, "An Adaptive Thermal Management Framework for Heterogeneous Multi-Core Processors,"*IEEE Transactions on Computers*, vol. 69, no. 6, pp. 894-906, 2020.

[159] D. Wang, K. Liu and X. Zhang, "Spatiotemporal Thermal Field Modeling Using Partial Differential Equations With Time-Varying Parameters," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 2, pp. 646-657, 2020.

[160] X. Kuang et al.,"Research on Control Strategy for a Battery Thermal Management System for Electric Vehicles Based on Secondary Loop Cooling," *IEEE Access*, vol. 8, pp. 73475-73493, 2020.

[161] X. Gong, J. Wang, B. Ma, L. Lu, Y. Hu and H. Chen, "Real-Time Integrated Power and Thermal Management of Connected HEVs Based on Hierarchical Model Predictive Control,"*IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 3, pp. 1271-1282, 2021.

[162] D. Gangadharan, J. Teich and S. Chakraborty, "Quality-aware video decoding on thermally-constrained MPSoC platforms,"*25th International Conference on Application-Specific Systems, Architectures and Processors*, 2014, pp. 256-263.

[163] S. Maity, A. Ghose, S. Dey and S. Biswas, "Thermal Load-aware Adaptive Scheduling for Heterogeneous Platforms," *33rd International Conference on VLSI Design and 2020 19th International Conference on Embedded Systems (VLSID)*, pp. 125-130,2020.

[164] K. R. Vaddina, J. M. Cebrián and L. Natvig, "Transient Temperature Prediction for Aging Thermal Sensors Using Artificial Neural Network," *24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pp. 51-57 2016.

[165] R. Sonavane, G. S. Kashyap and S. Chattopadhyay, "Thermal Aware Application Mapping and Frequency Scaling for Mesh-Based Network-On-Chip Design," *IEEE*

International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS), pp. 70-75, 2018.

[166] P. Meloni et al., "System Adaptivity and Fault-Tolerance in NoC-based MPSoCs: The MADNESS Project Approach," *15th Euromicro Conference on Digital System Design*, pp. 517-524. 2012.

[167] R. Mahajan et al., "Embedded Multidie Interconnect Bridge—A Localized, High-Density Multichip Packaging Interconnect," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 9, no. 10, pp. 1952-1962, 2019.

[168] P. Kaur and M. Agnihotri, "Efficient variable neighborhood search performance based joint optimization task allocation for multicore processor," *2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pp. 745-751, 2016.

[169] W. Zhang, H. Zhang and J. Lach, "Extending Performance-Energy Trade-offs Via Dynamic Core Scaling," *IEEE Transactions on Computers*, vol. 70, no. 11, pp. 1875-1886, 2021.

[170] X. Wang, A. K. Singh, B. Li, Y. Yang, H. Li and T. Mak, "Bubble Budgeting: Throughput Optimization for Dynamic Workloads by Exploiting Dark Cores in Many Core Systems," *IEEE Transactions on Computers*, vol. 67, no. 2, pp. 178-192, 2018.

[171] Y. Aslan, J. Puskely, J. H. J. Janssen, M. Geurts, A. Roederer and A. Yarovoy, "Thermal-Aware Synthesis of 5G Base Station Antenna Arrays: An Overview and a Sparsity-Based Approach," *IEEE Access*, vol. 6, pp. 58868-58882, 2018.

[172] G. Du et al., "Work-in-progress: SSS: self-aware system-on-chip using static-dynamic hybrid method," *International Conference on Compilers, Architectures and Synthesis For Embedded Systems (CASES)*, pp. 1-12, 2017.

[173] Borin, Lais, George Lima, Márcio Castro, and Patricia DM Plentz, "Dynamic power management under the RUN scheduling algorithm: a slack filling approach," *Real-Time Systems*, vol. 57, no. 4,pp. 443-484,2021.

[174] Park, Jurn-Gyu, Nikil Dutt, and Sung-Soo Lim, "An Interpretable Machine Learning Model Enhanced Integrated CPU-GPU DVFS Governor," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 6, pp.1-28,2021.

[175] Lee, Youngmoon, "Thermal-Aware Design and Management of Embedded Real-Time Systems," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1252-1255, 2021.

[176] Janniekode, Uma Maheshwar, Rajendra Prasad Somineni, Osamah Ibrahim Khalaf, Malakeh Muhyiddeen Itani, J. Chinna Babu, and Ghaida Muttashar Abdulsahib, "A Symmetric Novel 8T3R Non-Volatile SRAM Cell for Embedded Applications." *Symmetry*, vol. 14, no. 4, pp. 142-155 2022.

[177] J. Puskely, J. H. J. Janssen, M. Geurts, A. Roederer and A. Yarovoy, "Thermal-Aware Synthesis of 5G Base Station Antenna Arrays: An Overview and a Sparsity-Based Approach," *IEEE Access*, vol. 6, pp. 58868-58882, 2018.

[178] Iskandar, Veronia, Cherif Salama, and Mohamed Taher, "Dynamic thread mapping for power-efficient many-core systems under performance constraints," *Microprocessors and Microsystems*, vol.15. no.10. pp. 115-127, 2022.

[179] Y. Li, Z. Zhang, L. Zhang and D. Niu, "Thread-Level Speculation: Review and Perspectives," *5th International Conference on Information Science and Control Engineering (ICISCE)*, pp. 1291-12952018.

[180] C. Chen, T. Liao and C. Hwang, "An Integrated Embedded System for Evaluating Performance Parameters of CMOS Image Sensor," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 8, pp. 2328-2330, 2012.

[181] H. B. Parekh and S. Chaudhari, "Improved Round Robin CPU scheduling algorithm: Round Robin, Shortest Job First and priority algorithm coupled to increase throughput and decrease waiting time and turnaround time," *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, pp. 184-187, 2016.

[182] Harki, Naji, Abdulraheem Ahmed, and Lailan Haji,"CPU scheduling techniques: A review on novel approaches strategy and performance assessment," *Journal of Applied Science and Technology Trends*, vol.1, no. 2, pp.48-55, 2020.

[183] C. Liu, F. Tang, Y. Hu, K. Li, Z. Tang and K. Li, "Distributed Task Migration Optimization in MEC by Extending Multi-Agent Deep Reinforcement Learning Approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 7, pp. 1603-1614, 2021.

[184] T. Kim, M. Al-Tarazi, J. -W. Lin and W. Choi, "Optimal Container Migration for Mobile Edge Computing: Algorithm, System Design and Implementation,"*IEEE Access*, vol. 9, no. 4, pp. 158074-158090, 2021.

[185] L. Yang, D. Yang, J. Cao, Y. Sahni and X. Xu, "QoS Guaranteed Resource Allocation for Live Virtual Machine Migration in Edge Clouds," *IEEE Access*, vol. 8, pp. 78441-78451, 2020.

[186] N. S. Dey and T. Gunasekhar, "A Comprehensive Survey of Load Balancing Strategies Using Hadoop Queue Scheduling and Virtual Machine Migration," *IEEE Access*, vol. 7, pp. 92259-92284, 2019.

[187] X. Xu, L. Yao, M. Bilal, S. Wan, F. Dai and K. -K. R. Choo, "Service Migration Across Edge Devices in 6G-Enabled Internet of Vehicles Networks," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1930-1937, 2022.

[188] P. Munk, B. Saballus, J. Richling and H. Heiss, "Position Paper: Real-Time Task Migration on Many-Core Processors," *The 28th International Conference on Architecture of Computing Systems. Proceedings*, pp. 1-4, 2015.

[189] Al-Tarazi, J. -W. Lin and W. Choi, "Optimal Container Migration for Mobile Edge Computing: Algorithm, System Design and Implementation,"*IEEE Access*, vol. 9, no. 4, pp. 158074-158090, 2021.

[190] Jejurikar R, Pereira and Gupta R,"Leakage aware dynamic voltage scaling for real time embedded systems." *The proceedings of the 41st annual design automation conference ACM*, pp. 275–80, 2004.

[191] S. Safari, M. Ansari, G. Ershadi and S. Hessabi, "On the Scheduling of Energy-Aware Fault-Tolerant Mixed-Criticality Multicore Systems with Service Guarantee Exploration," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 10, pp. 2338-2354, 2019.

[192] Z. Shen and V. Dinavahi, "Real-Time MPSoC-Based Electrothermal Transient Simulation of Fault Tolerant MMC Topology," *IEEE Power & Energy Society General Meeting (PESGM)*, pp. 1-1, 2020.

[193] Marvell PXA270 Processor Electrical, mechanical and Thermal Specification data Sheet, http://www.marvell.com/application-processors/pxafamily/assets/pxa-27x-emts.pdf,2009.

[194] U. Richard, A. M. Deplanche and Y. Trinquet, "Storm a simulation tool for real-time multiprocessor scheduling evaluation," *IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA)*, pp. 1-8, 2010.

[195] Ouni, Bassem, Imen Mhedbi, Chiraz Trabelsi, Rabie Ben Atitallah, and Cécile Belleudy,"Multi-level energy/power-aware design methodology for MPSoC," *Journal of Parallel and Distributed Computing*, vol. 100, no.2 pp. 203-215,2017.