HAMLET: Adaptive Machine Learning-based Thermal Management in Fault-Tolerant Heterogeneous Multicore Systems

***Abstract*—The rapid advancement of technology and the increasing integration of multiple cores on a single chip have made managing power consumption and ensuring thermal safety critical challenges in real-time embedded systems. Increasing power densities have heightened the risk of overheating, leading to faults and system failures. This paper proposes a method (HAMLET) to manage power consumption, timing requirements, and reliability targets in multicore embedded systems through a genetic algorithm. The HAMLET balances efficient task mapping, energy-saving approach, and task replication to improve reliability. Moreover, predictive temperature control is achieved by a Long Short-Term Memory (LSTM) technique in the runtime phase, which dynamically adjusts task scheduling based on real-time temperature predictions. This ensures that tasks are mapped to cores in a way that prevents thermal violations while maintaining system reliability. Further, the HAMLET exploits the Dynamic Voltage and Frequency Scaling (DVFS) technique during static, pseudo-dynamic, and dynamic slack times to reduce power consumption and employs the Dynamic Power Management (DPM) technique to avoid thermal threshold violations. Our experimental results demonstrate that the proposed method significantly reduces peak power consumption and improves thermal management while maintaining the schedulability and reliability of hard real-time tasks. Compared to state-of-the-art techniques, our approach achieves up to 93.12% (on average 30.42%) reduction in peak power consumption and up to 155.31 % (on average 72.63%) reduction in energy consumption. Additionally, the schedulability is improved by up to 28.7% (on average 11.59 %) while satisfying the system's reliability targets.**

***Index Terms*—Power consumption, Genetic Algorithm, Long Short-Term Memory, Temperature prediction, Multicore embedded systems.**

I. INTRODUCTION

With the continuous advancement of CMOS technology, multicore platforms have become integral components of real-time embedded systems, providing high performance at relatively low costs. Reducing the size of CMOS transistors has accelerated the trend toward integrating more cores and electronic components onto a single chip, increasing computational capacity [1]–[4]. However, this increase in complexity manifests in reduced capacitance, higher transistor density, lower supply voltages, and elevated operating frequencies that present new challenges, especially decreasing system reliability and increasing power consumption and temperature [1], [2], [5]. Multicore embedded systems have become one of the most critical and widely adopted computing platforms in many different industries such as automotive, aviation, smart urban traffic control, and medical devices. The increasing range of applications confirms their growing significance, particularly

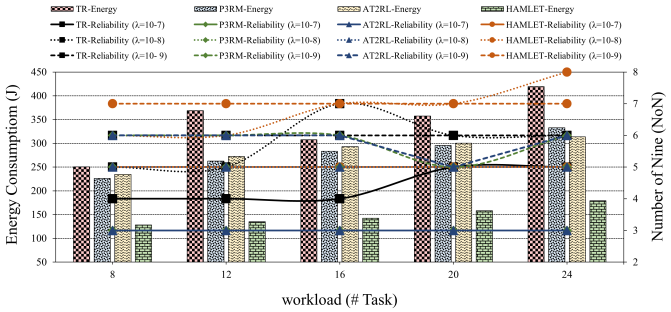

Fig. 1: A motivational observation of energy consumption and reliability comparison for different methods under varying workloads and fault rates.

in safety-critical fields such as healthcare, avionics, smart medical technologies, and smart grid, where system reliability is necessary for guaranteeing safety while meeting timing and power constraints to maintain system efficiency. As multicore chips continue to evolve, power consumption becomes an increasingly significant concern [2], [6]. The high power density directly causes higher chip temperatures which can reduce system reliability. To address these issues, effective power and thermal management techniques are required to guarantee performance and reliability [3], [4], [7]. While multicore systems inherently offer substantial potential for fault tolerance due to their inherent redundancy, the implementation of fault tolerance techniques often leads to increased power consumption [3], [6], [8], [9]. This necessitates an approach to managing power and reliability, aiming to minimize overheads simultaneously [2], [3], [10]. Real-time embedded systems, particularly those in safety-critical applications, must be capable of tolerating transient and permanent faults [3], [5], [8]. The increasing complexity and density of components in real-time embedded systems elevate the probability of both faults, making it essential to design systems capable of tolerating them simultaneously. Improving system reliability often increases power consumption, leading to challenges like thermal issues and performance degradation [2], [5]. Real-time embedded systems must balance three key goals: accurate task execution, meeting timing constraints, and managing power consumption. Any violation of these conditions can significantly compromise system reliability, leading to higher fault rates. Fault tolerance is typically achieved through various forms of redundancy, such as hardware, software, time, and information redundancy [3], [5], [8]. However, these

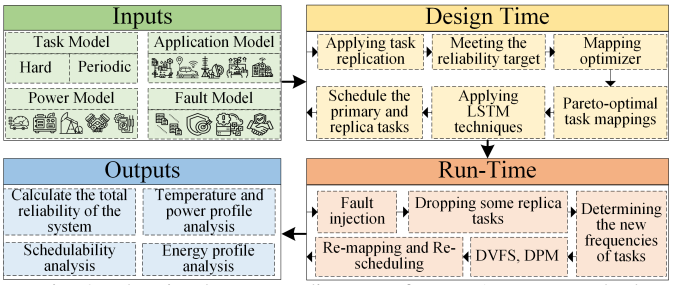

Fig. 2: The simple system diagram of our HAMLET method

techniques often introduce additional overhead, which must be carefully managed during system design. While multicore platforms inherently support fault tolerance due to their built-in redundancy, applying these techniques can increase power consumption, potentially exceeding safe temperature thresholds. To address these challenges, designers use approaches such as power and thermal-aware task scheduling, Dynamic Voltage and Frequency Scaling (DVFS), and Dynamic Power Management (DPM). These techniques illustrate the need to balance performance, reliability, power consumption, and timing constraints to achieve a sustainable, efficient operation in embedded systems [3], [7], [11], [12]. Fig. 1 compares energy consumption and reliability across four methods [5]—TR, [2]-P3RM, [4]-AT2RL, and HAMLET under varying workloads and fault rates (λ values of 10^{-7}, 10^{-8}, and 10^{-9}). HAMLET consistently maintains a high level of reliability, achieving over six "nines" across all fault rates, while also demonstrating the lowest energy consumption, particularly as workloads increase.

This paper presents HAMLET, a method designed to reduce energy consumption while meeting timing and reliability requirements in real-time embedded multicore systems. HAMLET employs an NSGA-II genetic algorithm (GA) that optimizes task mapping with three objectives: maximizing core utilization, minimizing power density by allocating tasks to low-power cores, and improving task allocation efficiency. This method ensures energy-efficient task distribution without compromising system performance. A task-replication technique is utilized before mapping and scheduling to maintain the reliability target. After the genetic algorithm identifies the optimal mapping during search space exploration, primary tasks are scheduled on Low-power cores using the Earliest Deadline First (EDF) algorithm, while replicated tasks are assigned to high-performance cores using the Earliest Deadline Later (EDL) algorithm. In the online phase, HAMLET dynamically adapts by switching to backup replicas if a primary task fails, thereby meeting timing constraints. This method uses the slack times through the DVFS technique, dynamically adjusting voltage and frequency to reduce power consumption on each core. HAMLET employs a Long Short-Term Memory (LSTM) neural network to predict future core temperatures and utilizes DPM Dynamic Power Management (DPM) to suspend task execution when temperatures exceed safe thresholds. These techniques prevent overheating and potential damage, ensuring sustainable operation under varying workloads. Fig. 2 shows the simple system diagram of our HAMLET method. The related work addresses various power, energy, and reliability management techniques in real-time embedded systems. The paper [6] presents two energy-efficient scheduling algorithms, BSESRG and BSESRG-FT, optimizing reliability, energy, and schedule length in heterogeneous systems. Techniques such as standby-sparing, and task replication focus on reliability improvement [2], [5]. Hybrid fault-tolerance (e.g., NMR with CVP) manages timing and power constraints [3].

Organization: Section II provides the details of our model and assumptions. Section III explains our Hamlet method. Section IV discusses the experimental evaluation and results. Finally, Section V presents the conclusion of the paper.

II. Models and Assumptions

This section introduces the task, system, power/energy, fault, and LSTM models.

A. System and Task Model

We consider in this paper a set of n independent periodic hard real-time tasks $\psi = \{T_1, \ldots, T_n\}$ such that each task T_i has a relative deadline D_i equal to its period π_i, an actual execution time t_i, a worst-case execution time wc_i at the maximum v/f level, and a finish time f_i for each task T_i. According to the periodic task model, the deadline for the j_{th} job of task T_i is given by $D_{ij} = j \times \pi_i$, while the arrival time of the job is represented as $t_{ij} = (j-1) \times \pi_i$. Each of the task's utilization is defined as $U_i = wc_i/\pi_i$ and the total task utilization is expressed as $U_{tot} = \sum_{i=1}^{n} U_i$. The reliability target in this paper refers to the desired and required level of system reliability for achieving specific objectives during system design and operation. To meet this target, we use a task replication technique.

B. Power and Energy Model

In this paper, we present a system model based on a multicore platform with m heterogeneous cores, denoted as $C = \{C_1, \ldots, C_m\}$. Each core in the system supports DVFS, enabling the adjustment of voltage and frequency across cores. This flexibility is necessary for reducing power consumption based on the specific workload and operational requirements of each core. The power consumption model for each core consists of two components:(1) static power (P_{static}), and (2) dynamic power ($P_{dynamic}$). The static power is primarily caused by leakage current. In contrast, dynamic power, which is generally the dominant component, varies according to the core's operational state and workload. The total power consumption for each core is expressed as follows [4], [6], [10], [11]:

$$P(V_i, f_i) = P_{static} + P_{dynamic} = I_0 e^{\frac{-V_{th}}{\eta V_{th}}} V_i + \alpha C_L V_i^2 f_i \quad (1)$$

The current I_0 is dependent on the technology and transistor size. The effective switching capacitance for each task is represented by C_L. Moreover, the parameters η, V_i, f_i, and α denote the technology-dependent factor, supply voltage,

operational frequency, and activity factor for each task, respectively. Given the linear relationship between voltage and frequency, we define the normalized voltage and frequency ρ_i as written: $\rho_i = V_i/V_{max} = f_i/f_{max}$. The system's total power consumption is the cumulative power consumption across all tasks and cores. This total power consumption can be expressed as [2], [3], [6]:

$$\begin{aligned} P_{total}(V_i, f_i) &= \rho_i(I_0 e^{\frac{-V_{th}}{\eta V_T}} V_{max} + \rho_i^3(\alpha_i C_L V_{max}^2 f_{max}) \\ &= \rho_i P_{static} + \rho_i^3 P_{dynamic} \end{aligned} \tag{2}$$

The total energy consumption of the system accounts for all jobs and tasks executed across cores. The total energy consumption is expressed as: [5], [11]:

$$E_{sys} = \sum_{k=1}^{m} \sum_{\forall T_{ij} \in core_m} \sum_{j=1}^{n_m} ((\rho_i P_{static} + \rho_i^3 P_{dynamic}).\frac{wc_{ij}}{\rho_i f_{max}}) \tag{3}$$

C. Fault Model

In this paper, we address both permanent and transient faults, with a primary focus on transient faults due to their higher frequency of occurrence. Transient faults are typically modeled using a Poisson distribution with an average arrival rate of λ, which characterizes the probability of fault occurrences over time. When DVFS is applied, the operational frequency decreases, leading to an increase in the fault rate. Consequently, the fault rate at frequency textitf is explained by the following model [2]–[4], [8]:

$$\lambda(f) = \lambda_0 10^{\frac{d(1-f)}{1-f_{mim}}} \tag{4}$$

where, λ_0 represents the transient fault rate at the maximum frequency f_{max} and d is a sensitivity factor that depends on both supply voltage and frequency scaling. Therefore, the reliability of task T_i at frequency f_i is calculated by the following equation [2]–[4]:

$$R_i(f_i) = e^{-\lambda(f_i)\frac{wc_i}{f_i}} \tag{5}$$

where, wc_i is the task T_i's worst case execution time, and $\lambda(f_i)$ is calculated by Eq. 5. For improved reliability, our proposed method uses the task replication technique. When k identical copies of a task T_i are executed on m different cores, the total reliability of the task is defined as the probability of at least one successful execution, given by [5], [10]:

$$R_{\text{total}}(T_i) = 1 - \prod_{j=1}^{k}(1 - R_j) \tag{6}$$

In summary, the total system reliability for n tasks running on the system is calculated as follows [2], [3], [5]:

$$R_{system} = \prod_{i=1}^{n} R_{total}(T_i) \tag{7}$$

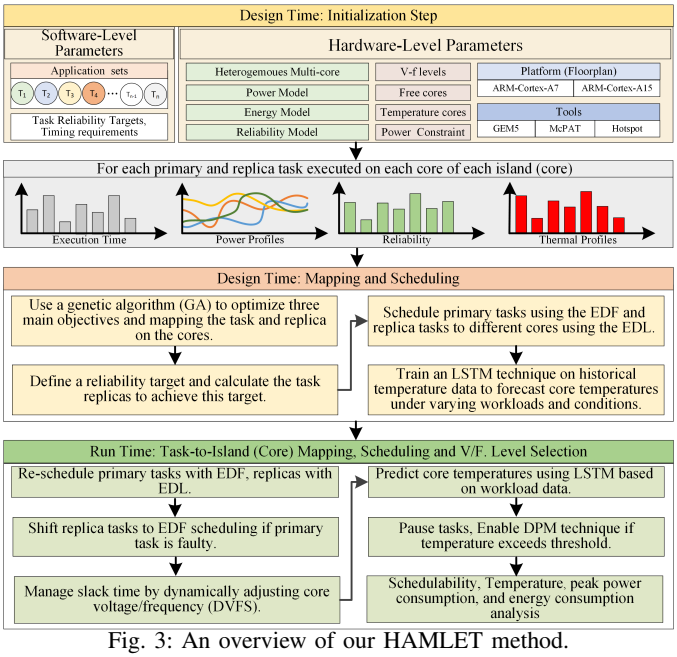

Fig. 3: An overview of our HAMLET method.

D. LSTM Model

The Long Short-Term Memory (LSTM) model for temperature prediction is designed to estimate the system's temperature at the next time step based on a sequence of recent temperature readings. Specifically, the model takes as input a sequence of temperature values, $\mathbf{X}_t = [T_i(t-1), T_i(t-2), \ldots, T_i(t-n)]$, obtained from the HotSpot tool, and predicts the core temperature $\hat{T}_i(t+1)$ at the subsequent time step. The model is trained by minimizing the Mean Squared Error (MSE) loss function, defined as:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{t=1}^{N}\left(\hat{T}_i(t+1) - T_i(t+1)\right)^2 \tag{8}$$

where $\hat{T}_i(t+1)$ is the predicted temperature and $T_i(t+1)$ is the actual temperature at time $t+1$. The LSTM architecture for temperature prediction consists of three main components: 1) the input gate, which stores incoming data into the cell state, 2) the forget gate, which filters and discards unnecessary information from past states, and 3) the output gate, which generates the final prediction based on the updated cell memory [13].

III. Power-Aware Fault Tolerance

A. Concept Overview

This paper focuses on a heterogeneous multicore system that executes periodic hard real-time tasks. The general problem of mapping and scheduling tasks on different types of processing cores is an NP-hard problem [3]. We propose a multi-objective genetic algorithm as the mapping optimizer. This solution decreases the peak power consumption while satisfying the reliability, timing, and power constraints.

B. Problem Definition

This paper proposes a method to manage power consumption, timing constraints, and reliability in heterogeneous

Algorithm 1 Task Mapping and Scheduling Algorithm

Input: Task set $\Psi = \{T_1, T_2, \ldots, T_n\}$, set of cores $C = \{C_1, C_2, \ldots, C_m\}$, power levels V/f, reliability target R_T, LSTM-based temperature predictor.
Output: The tasks mapping and scheduling on each core

1: Initialize $tasks\_queue$ $\mathcal{Q} \leftarrow \Psi$;
2: Set reliability target R_T for all tasks;
3: **while** $\mathcal{Q}$ is not empty **do**
4: **for** each task $T_i \in \mathcal{Q}$ **do**
5: Add replicas for T_i to meet R_T;
6: $C_{j,LP} \leftarrow T(i)_{Primary}.Map()$;
7: $C_{j,HP} \leftarrow T(i)_{Replica}.Map()$;
8: $T_i.vfLeve()$;
9: $R_{total}(T_i).Update$;
10: **end for**
11: **for** each core C_j **do**
12: Schedule primary tasks on LP core C_j with EDF;
13: Schedule replica tasks on HP core C_j with EDL;
14: **end for**
15: **for** each completed task T_i **do**
16: **if** fault occurs in T_i **then**
17: Switch to the replica of T_i;
18: $R(T_i).Update()$ and $tasks\_queue$;
19: **end if**
20: Remove T_i from $\mathcal{Q}$;
21: **end for**
22: **Thermal Management:**
23: **if** $T_{LSTM\_predicted} > T_{DTM}$ **then**
24: $Active.DPM(T_i)$;
25: **if** core temperature is safe **then**
26: $Resume.T_i()$;
27: **end if**
28: **end if**
29: $power(T_i).Update()$ and $R_{total}(T_i).Update()$;
30: **end while**

multicore systems. Since these parameters inherently conflict with each other, our objective is to achieve a carefully balanced trade-off. To achieve this, the HAMLET method employs the NSGA-II genetic algorithm to optimize three main objectives: 1) maximize core utilization on low-power (LP) cores, 2) minimize power density across the system, and 3) maximize executing the primary tasks on LP cores and replica tasks on high-performance (HP) cores for reducing power consumption. To achieve high reliability, our proposed method uses the task replication technique. We set a reliability target and determined the required number of task replicas to meet this target. Tasks and their replicas are mapped onto cores with the lowest utilization mapping policy and the primary tasks are scheduled on LP cores by the Earliest Deadline First (EDF) policy and replica tasks are scheduled on HP cores with the Earliest Deadline Later (EDL) policy. At runtime, if conditions necessitate replica execution, these tasks are immediately scheduled with EDF instead of EDL. When a primary task is faulty, the system switches to the corresponding replica task under EDF scheduling policy. To further reduce power consumption, HAMLET dynamically manages slack time using the DVFS technique. By creating slack times between the execution of primary tasks and dropping unnecessary replica tasks at runtime, the system adjusts the voltage and frequency levels of the cores, thereby reducing energy consumption. Proactive thermal management is achieved through a Long Short-Term Memory (LSTM) neural network, which predicts core temperature trends. If the Dynamic Thermal Management (DTM) threshold temperature (T_{DTM}) is exceeded, Dynamic Power Management (DPM) switches cores into low-power states, resuming normal operation once safe temperature levels are restored. The overview of our proposed HAMLET method is shown in Fig. 3. In the below notations, our objective is to reduce power consumption and our constraints are power, reliability target, and timing requirements. In the following, the general problem is defined:

Optimization Objective: The total power consumption, calculated as the sum of the power consumed by all jobs of both primary and replica tasks, should be minimized.

$$Minimize\ P_{total}(V_i, f_i) = \rho_i P_{static} + \rho_i^3 P_{dynamic} \tag{9}$$

Task to Core Assignment Constraint: Each primary task must execute on one LP core unless its corresponding replica task is mapped to an HP core. An element $X_{ihl} = 1$ indicates that task i is mapped to core h and executes under v/f level l.

$$\forall i, l : \sum_{h} X_{ihl} = 1 \tag{10}$$

Reliability Target: The reliability of each task $R(T_{i,j})$ should be improved through the task replication fault-tolerant technique to meet the reliability target.

$$\forall i, j : R(T_{i,j}) \geq R_{target} \tag{11}$$

Timing Constraints: The finishing time of each task, f_{T_i}, must not exceed its deadline.

$$f_{T_i} \leq D_i \tag{12}$$

Uilization Constraint: The total utilization of the system after setting the voltage and frequency level of the tasks must not exceed the number of available cores. Additionally, the combined utilization of all tasks assigned to any single core must remain below 1.

$$U_{tot} \leq m \tag{13}$$

C. Algorithm Discussion

We propose a mapping and scheduling policy for hard real-time systems that simultaneously satisfies timing, reliability, and power constraints. The algorithm reduces power consumption by preventing peak power overlaps during concurrent task execution and dynamically adjusting the voltage/frequency (v/f) levels of tasks based on slack time. The task mapping and scheduling algorithm organizes tasks by initializing a queue (line 1) and setting a reliability target for each task to improve system reliability (line 2). In lines 5-7, each task receives replicas to meet the reliability target, with primary and replica tasks mapped to cores through a genetic NSGA-II optimization algorithm to reduce energy consumption. Voltage and frequency levels are adjusted based on available slack times (DVFS), further reducing power consumption in line

TABLE I: The details of system configuration

Parameter	Island Configuration	
	Low Power	**High Performance**
Core Type	ARM Cortex-A7	ARM Cortex-A15
Machine Type	In-Order	Out-Of-Order
V/f level	[0.9V, 0.8GHz] to [1.1V, 1.6GHz]	[0.9V, 1GHz] to [1.1V, 2GHz]
Microarchitecture	ARMv7-A	ARMv7-A
L1 Cache	32KB, 8KB block-width, 2-way	32KB, 8KB block-width, 2-way
L2 Cache	1MB, 8-way	1MB, 16-way
Memory	1GB, 32-bit LPDDR3e	2GB, 32-bit LPDDR3e

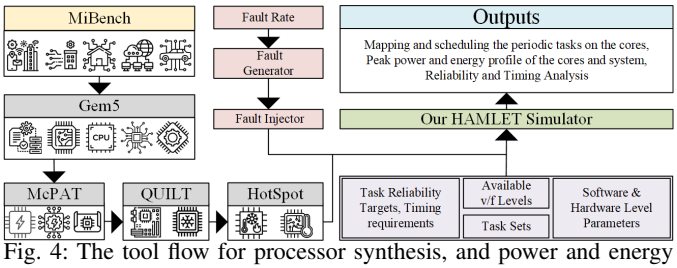

Fig. 4: The tool flow for processor synthesis, and power and energy evaluation

8. In lines 12-13, the system schedules primary tasks with the Earliest Deadline First (EDF) on LP cores and replica tasks with the Earliest Deadline Later (EDL) on HP cores. When faults are detected, the system switches execution from primary tasks to replica tasks (lines 16–17). An LSTM-based predictor manages core temperatures by activating Dynamic Power Management (DPM) if temperatures exceed the threshold temperature (T_{DTM}) in lines 23-26. In line 29, Reliability and power consumption are updated iteratively to ensure that all tasks are successfully scheduled. Algorithm 1 maps and schedules a set of N tasks, including primary and replica tasks, across C cores over H time slots. The time complexity of Algorithm 1 is O($N^{\psi} \times C \times H$), where ψ includes different parameters such as frequency levels.

IV. Experimental Evalutation

This paper introduces an advanced HAMLET method to balance task scheduling, reliability, and power consumption in heterogeneous multicore systems. The task model is characterized by periodic hard real-time requirements and satisfies level C of the DO-178B standard, ensuring applicability in safety-critical systems. The evaluation considers heterogeneous multicore systems with varying numbers of cores (i.e., $m \in [4, 8, 16, 32]$). Real-life applications from the Mibench benchmark suite are used for comprehensive tests for real-world applicability. The experimental setup features ARM Cortex-A7 and ARM Cortex-A15 cores, chosen for their support of per-core Dynamic Voltage and Frequency Scaling (DVFS), which is crucial for reducing energy consumption and managing core temperatures. Details of different architectural configurations are summarized in Table I. We use the gem5 full-system simulator for detailed system modeling, McPAT simulator for power consumption analysis of the tasks, QUIL simulator to generate the chip floorplan based on the results of McPAT, and Hotspot simulator for temperature modeling to evaluate our HAMLET method's efficacy. Figure 4 provides an overview of the tools utilized during the evaluation. The evaluation employs a fault injection model, where a fault vector is generated for each processing core to simulate real-world fault scenarios. We use Eq. 4 with parameters $\lambda_0 = 10^{-7} faults/\mu s$ (to inject faults), and $d = 2$ to generate fault vectors that depend on voltage and frequency levels. We compare our HAMLET method with state-of-the-art methods, specifically [5]-TR, [4]-AT2RL, and [2]-P3RM. Each method is evaluated for schedulability, reliability, peak power consumption, and energy savings, enabling a comparative analysis that shows the advantages of our HAMLET method.

A. Evaluating Schedulability

guaranteeing real-time tasks meet their deadlines is a critical requirement for real-time embedded systems. We evaluated the schedulability of the HAMLET method and compared its performance with state-of-the-art methods, as shown in Fig. 5(a). Our method demonstrates a significant focus on the real-time characteristics of the system, maintaining 100% of schedulability. Specifically, the HAMLET improves schedulability by an average of 11.5%, with improvements reaching up to 28.7%.

B. Evaluating the Peak Power Consumption

Power management is critical for multicore systems as it directly impacts system performance and reliability. The peak power consumption of HAMLET was compared with [5]-TR, [4]-AT2RL, and [2]-P3RM methods under different system configurations with 4, 8, 16 and 32 cores. As shown in Fig. 5(b), HAMLET consumes significantly less peak power than the other methods and does not violate the power constraints. Specifically, HAMLET reduces peak power by an average of 30.42% and up to 93.12% compared to state-of-the-art methods across different cores.

C. Evaluating the Energy Saving

A detailed comparison of energy consumption for HAMLET and other methods is presented in Fig. 5(c). The results demonstrate that HAMLET achieves superior energy efficiency across all evaluated configurations (4, 8, 16, and 32 cores). Also, Fig. 5(c) shows that energy consumption increases with higher workload and core utilization,

D. Evaluating Reliability at Runtime

The total reliability of the system for our HAMLET method compared to the state-of-the-art methods is presented in this subsection. The reliability of methods was analyzed using the Number of Nines (NoN) metric shown in Fig. 5(d). Note that our HAMLET method has replica tasks that can tolerate both transient and permanent faults. It is inferred that in Fig. 5(d) for all numbers of cores, the HAMLET method consistently achieves higher reliability.

Among other methods, [5]-TR performs slightly better than [4]-AT2RL and [2]-P3RM, but all fall short of HAMLET's reliability.

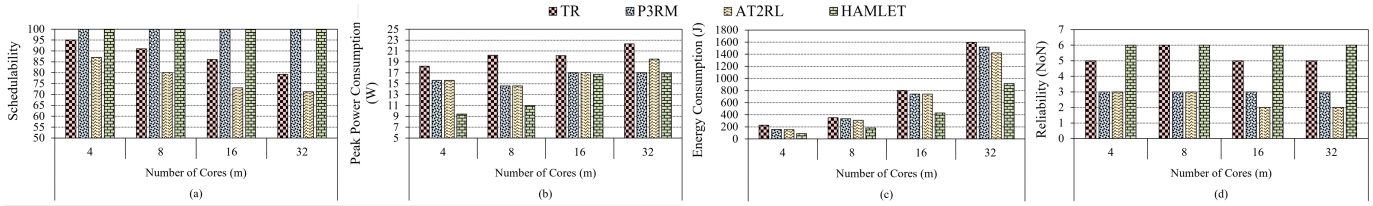

Fig. 5: Comparison between HAMLET and state-of-the-art methods. a) Schedulability b) Peak power consumption, c) Energy consumption, d) The total reliability of the system at runtime

E. Evaluating the Temperature Management

The performance evaluation of the trained LSTM model uses key metrics to assess prediction accuracy, error variability, and robustness. The Mean Absolute Error (MAE), calculated as $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|T_i - \hat{T}_i\right|$, measures the average magnitude of errors. The Mean Absolute Percentage Error (MAPE) captures the relative error as a percentage of actual temperature, expressed as $\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{T_i - \hat{T}_i}{T_i}\right|$, provides relative errors as a percentage. To capture error variability, the Standard Deviation of Absolute Error (SDAE) is calculated as $\text{SDAE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\left|T_i - \hat{T}_i\right| - \text{MAE}\right)^2}$. Lastly, the Probability of Outlier Error (POE) quantifies extreme errors, computed as $\text{POE} = \frac{\text{Number of Outlier Errors}}{n}$. Model accuracy approximated as $\text{Accuracy} = 100 - \text{MAPE}$. These metrics provide a comprehensive evaluation of the LSTM model's ability to predict future core temperatures for efficient thermal management. The results are reported in Table II. Fig. 6 presents the model's accuracy and loss functions, showing a decrease in accuracy as workload and the number of cores increase.

V. CONCLUTIONS

HAMLET is a novel method that addresses power, timing, and reliability challenges in real-time multicore embedded systems. By using a multi-objective NSGA-II genetic algorithm, HAMLET optimizes task mapping to maximize core utilization, minimize power density, and enhance system efficiency. Reliability is improved through task replication, energy savings are achieved using Dynamic Voltage and Frequency Scaling (DVFS), and thermal management is ensured through LSTM-based temperature predictions combined with Dynamic Power Management (DPM). Tasks are managed to meet system reliability targets while satisfying task-level timing and power constraints. Experimental results demonstrate that HAMLET reduces peak power consumption by an average of 30.42% and improves schedulability by 11.5% compared to state-of-the-art methods, highlighting its effectiveness in balancing reliability and efficiency in multicore systems.

TABLE II: Evaluation Metrics for LSTM-Based Temperature Prediction

# of Cores	MAE	MAPE (%)	MSE	SDAE	POE (%)
4	0.85	2.85	1.96	0.45	3.2
8	0.92	3.10	2.15	0.51	4.1
16	0.96	3.35	2.39	0.58	5.0
32	0.98	3.55	2.52	0.62	5.8

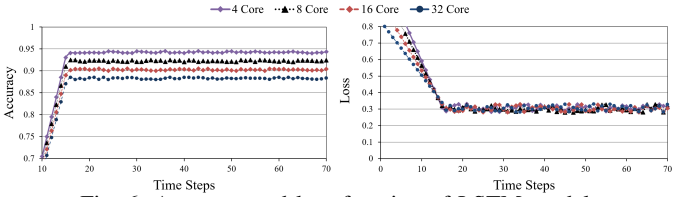

Fig. 6: Accuracy and loss function of LSTM model.

REFERENCES

[1] I. Moghaddasi, M. E. Salehi Nasab, and M. Kargahi, "Aging-aware instruction-level statistical dynamic timing analysis for embedded processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 2, pp. 433–442, 2020.

[2] S. Yari-Karin, R. Siyadatzadeh, M. Ansari, and A. Ejlali, "Passive primary/backup-based scheduling for simultaneous power and reliability management on heterogeneous embedded systems," *IEEE Transactions on Sustainable Computing*, vol. 8, no. 1, pp. 82–93, 2022.

[3] A. H. Ansari, M. Ansari, and A. Ejlali, "Taft: Thermal-aware hybrid fault-tolerant technique for multicore embedded systems," *IEEE Embedded Systems Letters*, pp. 1–1, 2024.

[4] S. M. P. Dinakarrao, A. Joseph, A. Haridass, M. Shafique, J. Henkel, and H. Homayoun, "Application and thermal-reliability-aware reinforcement learning based multi-core power management," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 15, no. 4, pp. 1–19, 2019.

[5] M. A. Haque, H. Aydin, and D. Zhu, "On reliability management of energy-aware real-time systems through task replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 3, pp. 813–825, 2016.

[6] S. K. Biswas, P. K. Muhuri, and U. K. Roy, "Binary search-based fast scheduling algorithms for reliability-aware energy-efficient task graph scheduling with fault tolerance," *IEEE Transactions on Sustainable Computing*, 2023.

[7] S. Chakraborty, Y. Sharma, and S. Moulik, "Treafet: Temperature-aware real-time task scheduling for finfet based multicores," *ACM Transactions on Embedded Computing Systems*, 2024.

[8] J. Zhou, X. S. Hu, Y. Ma, J. Sun, T. Wei, and S. Hu, "Improving availability of multicore real-time systems suffering both permanent and transient faults," *IEEE Transactions on Computers*, vol. 68, no. 12, pp. 1785–1801, 2019.

[9] W. Fan, F. Xiao, M. Lv, L. Han, and S. Yu, "Efficient fault-tolerant path embedding for 3d torus network using locally faulty blocks," *IEEE Transactions on Computers*, pp. 1–14, 2024.

[10] L. Niu and D. Zhu, "Fixed-priority scheduling for reliable and energy-aware (m, k)-deadlines enforcement with standby-sparing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 3, pp. 502–515, 2022.

[11] A. Roy, H. Aydin, and D. Zhu, "Energy-aware standby-sparing on heterogeneous multicore systems," in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2017, pp. 1–6.

[12] J. Geng, Z. Zhu, W. Liu, X. Zhou, and B. Li, "PowerLens: An adaptive DVFS framework for optimizing energy efficiency in deep neural networks," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*. New York, NY, USA: ACM, Jun. 2024, pp. 1–6.

[13] I. Syafalni, C. Amadeus, N. Sutisna, and T. Adiono, "Efficient real-time smart keyword spotting using spectrogram-based hybrid cnn-lstm for edge system," *IEEE Access*, vol. 12, pp. 43 109–43 125, 2024.