Ring-DVFS: Reliability-Aware Reinforcement Learning-Based DVFS for Real-Time Embedded Systems

Amir Yeganeh-Khaksar, Mohsen Ansari, Sepideh Safari, Sina Yari-Karin, and Alireza Ejlali

***Abstract*—Dynamic Voltage and Frequency Scaling (DVFS) is one of the most popular and exploited techniques to reduce power consumption in multicore embedded systems. However, this technique might lead to a task-reliability degradation because scaling the voltage and frequency increases the fault rate and the worst-case execution time of the tasks. In order to preserve task-reliability at an acceptable level as well as achieving power saving, in this letter, we have proposed an enhanced DVFS method based on reinforcement learning to reduce the power consumption of sporadic tasks at runtime in multicore embedded systems without task-reliability degradation. The reinforcement learner takes decisions based on the power savings and task-reliability variations due to DVFS and considers the suitable voltage-frequency level for all tasks such that the timing constraints are met. Experimental evaluation was done on different configurations and with different numbers of tasks to investigate the efficiency of the proposed method. Our experiments show that our proposed method works efficiently than other existing works for reducing power consumption without reliability degradation and deadline misses.**

***Index Terms*—Power Management, DVFS, Sporadic Tasks, Reliability, Reinforcement Learning, Multicore Platforms.**

I. Introduction

MULTICORE embedded systems, coupled with increased power consumption, pose multiple challenges such as reliability and performance [1][2]. In order to overcome such concerns, exploiting the reliability-aware power management technique is a crucial requirement for multicore embedded systems [1][8]. Towards meeting the power constraints, the Dynamic Voltage and Frequency Scaling (DVFS) technique is one of the most effective and widely exploited techniques in the mentioned systems [9]. Since most of multicore embedded systems are employed for executing real-time tasks [8], exploiting DVFS might lead to missing task deadlines, and this is not acceptable in real-time embedded systems [7]. Advancements in the machine learning (ML) domain led to its adoption for prediction of the parameters of DVFS, using different techniques such as reinforcement learning (RL), regression analysis, etc. For example, the authors in [17] have introduced a deep RL method that tries to manage thermal and power issues at runtime by efficiently allocating resources and distributing tasks to the cores. The main purpose is to maximizing performance per watt (PPW) by determining performance-energy configurations, which means the number and frequency of big.LITTLE cores in the processors. The authors in [18] have proposed a deep RL-based reliability management method that considers soft and hard errors. The proposed method in [18] is based on physics-based three-phase electromigration and exponential soft error models, and it reduces memory consumption and computational time significantly compared to the state-of-the-art RL-based methods. In [19], the authors employ RL to introduce a new task mapping method, called LifeGuard, to address performance and aging issues. The LifeGuard works according to the performance requirements of applications and their aging behaviors. This method learns and finds the most appropriate core (from the lowest frequency to the highest frequency) for different types of applications. The authors in [5] illustrated a power management method that performs RL to manage power consumption without the application- and thermal-reliabilitiy degradation. In this method, the learner tries to find suitable VF settings for the next execution time of each application and feed them to the system at run-time. Since none of the related work considers real-time constraints, this paper proposes a reliability-aware reinforcement learning (RL)-based power management technique for multicore real-time embedded systems such that all timing constraints are met.

In this letter, we provide the following novel contributions: (*i*) To the best of our knowledge; this is the first work that proposes reliability-aware reinforcement learning (RL)-based power management technique for multicore real-time embedded systems, (*ii*) The proposed technique determines the VF level for each task such that the power, timing, and reliability constraints are met simultaneously.

II. Models and Assumptions

A. Application, Hardware, Reliability, and Power Model

Application Model: We have considered a set of N independent soft sporadic tasks, denoted as $T = \{\tau_1, \tau_2, ..., \tau_N\}$. Each task τ_i can be illustrated by a triple $\tau_i = (w_i, d_i, p_i)$, where w_i is the worst-case execution time, d_i is the relative deadline, and p_i is the minimum inter-arrival time (or period). In this work, we have considered that $d_i \le p_i$.

Hardware Architecture Model: We have considered a processor with M homogeneous cores, denoted as the set of $C = \{c_1, c_2, ..., c_M\}$. Each core c_i can execute tasks on k different voltage-frequency (VF) levels from (v_1, f_1) to (v_k, f_k). We have assumed that the cores operating at higher VF levels consume more power, and hence, have higher performance [1].

Manuscript received Oct. 4, 2020; accepted Oct. 20, 2020. This work was supported by the Sharif University of Technology. This paper was recommended by Associate Editor A. Kumar. (*Corresponding author: Alireza Ejlali*.)

The authors are with the Department of Computer Engineering, Sharif University of Technology, Tehran 14588, Iran (e-mails: {ayeganeh; mansari; ssafari, and sinayari}@ce.sharif.edu; ejlali@sharif.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier x.y/z

Also, we have assumed that idle cores go to sleep mode to reduce dynamic power consumption [1][8].

Reliability Model: In this letter, we have considered transient faults. In most cases, transient faults are modeled as a Poisson distribution using the average fault rate of λ [1][7]. The average fault rate depending on the voltage of the processing core is calculated as follows [1][5][7][8]:

$$\lambda(V) = \lambda_0 10^{(V_{\max}-V)/\Delta} \quad (1)$$

As can be seen, decreasing voltage increases the fault rate exponentially [1]. Also, the maximum stable frequency for a core is limited by its supply voltage [2]. Therefore, reducing supply voltage and operating frequency using DVFS increases the fault rate λ significantly [1]. In Eq. (1), Δ and λ_0 are sensitivity factor to voltage changes and fault rate at maximum frequency, respectively. We have assumed $\Delta = 1$ volt and $\lambda_0 = 10^{-7}$ faults/μsec [7]. Also, Eq. (2) shows the reliability model of the task τ with the actual execution time t in supply voltage V in which $\lambda(V)$ is given by Eq. (1) [1][5][7][8].

$$R(\tau, V) = e^{-\lambda(V)t} \quad (2)$$

Power Model: The total power consumption of each core consists of dynamic and static components [1][7][8]. System activity and leakage power are the main reasons for dynamic and static power consumption, and the total power consumption is modeled by the following equation [1][7][8][12]:

$$P(V, f) = P_{dynamic} + P_{static} = \alpha C V^2 f + I_0 e^{\frac{-V_{th}}{\eta V_T}} V \quad (3)$$

Where α is the switching activity factor, C is the average capacitance, I_0 and η are technology parameters, V_T is the thermal voltage, and V_{th} is the threshold voltage [1][7][8].

B. Reinforcement Learning (RL)

Reinforcement Learning (RL) is a machine learning technique that enables an agent to learn through trial-and-error in an uncertain or dynamic environment and to gain experience from its own previous actions [3][4][5]. The agent should select sequential actions in such a way that the total future reward is maximized [3][4][5]. The interactions between agent and environment are modeled using a finite state space S, a set of available actions A, and a future reward function $R: S \times A \rightarrow R$ [9]. The policy π is a map from state s to action a $(\pi: S \rightarrow A)$, and its stochastic form is $\pi(a \mid s) = \mathrm{P}[A_t = a \mid S_t = s]$ [9][10].

Q-Learning: Q-learning is one of the most popular algorithms exploited to perform RL [3][4]. This is a value-based RL algorithm that is exploited on the Q-value denoted as Q(s, a) for each state-action pair, and its value approximates the expected long-term cumulative discounted reward of taking action a starting from state s, using temporal difference [5][10]. Q-values are updated when an action is issued, and the corresponding reward is received [5]. The Q-value is equal to the sum of the actual current reward and discounted estimated future value that can be computed as follows [5][9][11]:

$$Q'(s,a) \leftarrow (1-\alpha) Q(s,a) + \alpha \left[R(s,a,s') + \gamma \max_{a' \in A} Q(s',a') \right] \quad (4)$$

Where $\alpha \in (0,1)$ is the learning rate, $R(s,a,s')$ is the expected reward after taking action a from state s, and $\gamma \in (0,1)$ is the discount factor. In Eq. (4), s/s' and $Q(s,a)/Q(s',a')$ are states and Q-states, respectively. $Q(s,a)$ represents the consideration of taking action a from state s, and its value, as mentioned earlier, is the expected long-term cumulative discounted reward. Therefore, the learner selects the maximum Q-value during the learning process. In our proposed method, we have assumed that α is fixed, like [5], and is equal to 0.75. Also, the discount factor is equal to 0.25. According to Eq. (4), $\pi(s) = \max_{a \in A} Q(s,a)$ is the agent's policy that selects the maximum Q-value at each time [5][11]. Also, Eq. (5) shows the optimal policy $\pi^*(s)$ based on Bellman's principle of optimality [11] when the current state is s and $Q'(s,a)$ is given by Eq. (4) [16].

$$\pi^*(s) = \arg\max_{a \in A'} \left(Q'(s,a) \right) \quad (5)$$

Where A' is a subset of A that determines the set of the appropriate actions (VF levels) for the corresponding task according to real-time constraints.

III. Problem Definition and Our Solution

A. Concept Overview

In this letter, we have proposed an enhanced DVFS method to reduce power consumption in a homogeneous multicore system that executes sporadic tasks without reliability degradation exploiting RL.

B. Problem Definition

State Space: There are several criteria for determining the state of the system [5][9]. In our proposed method, the state of the system for the agent is the per-task power consumption and reliability based on Eq. (3) and Eq. (2), respectively. Therefore, we have considered $S = \{(\tau, (p_i, r_i)) \mid \forall \tau \in RT, RT \subseteq T, i \in [1,n]\}$ as a set of states, where RT is a subset of T that determines the set of the running tasks, n is the number of quantized power levels, and each state $s_{\tau,i} = (\tau, (p_i, r_i))$ represents the power consumption of the running task τ and its reliability. Also, the states sorted by the power consumption in ascending order.

Action Space: We have assumed there are k actions, denoted as the set of $A = \{a_1, a_2, ..., a_k\}$. Each action a_i illustrates assigning i-th VF level to the task, i.e., $a_i = (v_i, f_i)$. As mentioned earlier in section II.A, the variable k indicates the number of available VF levels. It should be noted that some VF levels cannot be applied to some tasks due to the reliability target. In this work, like [5], we have considered only six VF levels (i.e., $k = 6$) to limit the number of feasible actions and reduce complexity and convergence issues.

Reward: A reward is a scalar feedback signal that indicates how well the agent did in the last step. In order to manage power consumption without reliability degradation, a reward should be calculated as a function of variations in power consumption and reliability simultaneously [5]. Similar to the work [5], we have overcome these concerns, i.e., power consumption and reliability, by using principal component analysis [6]. In other words, the calculation of the reward of the transition of task τ from state s with action a to state s' (all possible states for task τ from state s with action a) is as follows [5]:

$$R(s,a,s') = \alpha_1 (\Delta P/P) + \alpha_2 (\Delta R/R) \quad (6)$$

Where $\Delta P/P$ and $\Delta R/R$ are the variations in power consumption and reliability due to transition. In this letter, reliability is based on Eq. (2), and also we have assumed $\alpha_1 = \alpha_2 = 0.5$.

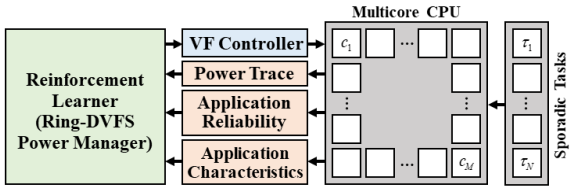

Fig. 1. The overview of the design flow of Ring-DVFS.

C. Power Predictor Policy

Due to computational issues of reactive power management, we have to predict power trace according to previous traces for efficient run-time power management, like [5]. This prediction is based on Eq. (7).

$$p(t+1) = \sum_{i=0}^{x} w_i p(t-i) + \varepsilon \quad (7)$$

Where $p(t+1)$ is the predicted power for time-slot $t+1$, w_i is the regression coefficient, and ε is the error. We have assumed $x = 4$ in our evaluations. After power prediction, in order to avoid complexity and convergence issues, the predicted power is quantized, and the state with the closest power consumption is selected as the current state. We have considered four states (i.e., $n = 4$) in this work.

D. Ring-DVFS Power Management

An overview of the design flow of Ring-DVFS is presented in Fig. 1. Also, we have provided an Algorithm 1 to show the pseudo-code of the Ring-DVFS power management method. First, in Line 1, the power trace of the next time-slot of an issued sporadic task is predicted through Eq. (7). Then, the corresponding reliability is estimated through (2) in Line 2. According to Lines 1-2 and the set of states, one of the states is selected in Line 3. In Lines 4-5, based on Bellman's principle of optimality, i.e., Eq. (5), maximum Q-value is selected as the action and fed to the VF controller. Finally, the reward is calculated based on Eq. (6), and Q-values are updated based on Eq. (4) in Lines 6 and 7, respectively. In our evaluations, to overcome the convergence issue, we have done Ring-DVFS at regular intervals to have enough time for making decisions, and also limited the number of iterations because using deep RL might increase computational and latency overheads.

IV. Results and Discussion

In this section, we evaluate the effectiveness of Ring-DVFS, employing a gem5 full-system simulator [16].

A. System Settings

We ran our simulations with various task sets, including real-life applications of the MiBench benchmark suite [13] running on a target homogeneous multicore platform. We considered that the system supports per-core DVFS. The details of simulation configurations for the processing cores of our system are summarized in Table 1. Then, we have connected the gem5 to a simulator that is written by Python.

B. Power Savings

We have compared our proposed method with other methods in terms of reducing power consumption. Fig. 2 shows the comparison of [5]-RLPM, [14]-Linear, [15]-STM, [17]-PPW, [18]-DRL, [19]-LG, and our proposed method. As can be seen,

Algorithm 1. Ring-DVFS Power Management

Input: Tasks' Power Traces, Reliabilities, and Characteristics
Output: VF Settings

start Ring-DVFS
1: Predict the power trace through Eq. (7).
2: Estimate corresponding reliability through Eq. (2).
3: Determine the state according to Lines 1-2.
4: Select an action (VF settings) through Eq. (5).
5: Feed VF settings to the VF controller.
6: Calculate the reward through Eq. (6).
7: Calculate the Q-values through Eq. (4).
end Ring-DVFS

by increasing the number of cores, RL-based methods have better performance in reducing power consumption. In other words, in small systems with a small number of cores (e.g., single- or dual-core), lightweight methods, like [14]-Linear, are more effective than the others. RL-based power management methods have more complexity and computations than power savings at runtime. Therefore, exploiting them in small systems are not efficient. The evaluations' results indicate that our proposed method provides, on average by –3.2%, 42.75%, 33.5%, –11.5%, –14.75%, and 9% average power reduction compared to [5]-RLPM, [14]-Linear, [15]-STM, [17]-PPW, [18]-DRL, and [19]-LG, respectively. The increased run-time power savings with [5]-RLPM, [17]-PPW, and [18]-DRL compared to our method is because of the real-time characteristics of our workloads.

C. Reliability

As stated in section III.B, the reward function considers both power savings and reliability variations in this work. Also, as mentioned earlier in section II.A, we have considered $\Delta = 1$ volt and $\lambda_0 = 10^{-7}$ faults/μsec. According to Fig. 3, the results show that our proposed method achieves higher reliability than other power management methods. In the situation of executing various sets of tasks based on MiBench benchmark suite on the 16-core processor, compared to the proposed method, [5]-RLPM, [14]-Linear, [15]-STM, [17]-PPW, [18]-DRL, and [19]-LG have on average by 7%, 59%, 50.1%, 93%, 6.5%, and 5.8% more variance in reliability, respectively. It should be noted that [5]-RLPM considers both application- and thermal-reliabilities' variations.

D. Overhead Analysis

Run-time power management techniques impose computation overheads (e.g., determining the appropriate VF settings and assigning them to the processing cores) to the system. In this letter, we have considered the additional execution time of each

Table 1. The details of system configuration

Parameter	Configuration
Core Type	ARM Cortex-A7
Machine Type	In-Order
Core Volt. And Freq.	Six different VF levels from [0.85Volt, 1GHz] to [1.1Volt, 2GHz]
The Number of Cores	4, 8, 12, and 16-core system
Core Microarchitecture	ARMv7-A
L1 Cache	32KB, 8KB block-width, 4-way
L2 Cache	2MB, 16-way
Memory	2GB, 32-bit LPDDR3e

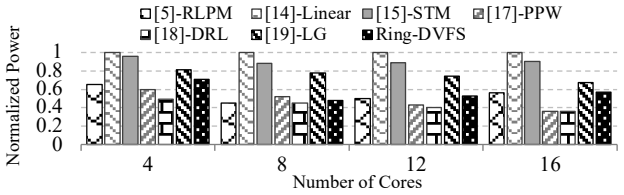

Fig. 2. Average power consumption comparison of different power management schemes.

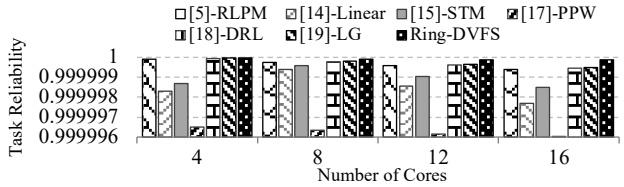

Fig. 3. Average task reliability comparison of different power management schemes.

Table 2. Average percentage of additional execution time for MiBench applications running on multicore system

[14]-Linear	[15]-STM	[17]-PPW	Ring-DVFS	[5]-RLPM	[19]-LG	[18]-DRL
47.5%	28.7%	26.9%	24.8%	22.7%	20.5%	19.1%

task as an overhead, and we have compared the average run-time of our proposed method with different power management techniques. Table 2 shows the average percentage of additional execution time for various sets of tasks based on MiBench benchmark suite running on 4-, 8-, 12-, and 16-core systems, and also under six different VF levels. Our evaluations show, in similar situations, [14]-Linear has more additional execution time than the others. Also, due to embedded learning based on tasks' characteristics, RL-based methods have less additional execution time than [14]-Linear and [15]-STM. Our proposed method has on average by 2.1%, 5.7%, and 4.3% more additional execution time than [5]-RLPM, [18]-DRL, and [19]-LG, respectively. It should be noted that these differences are mainly because of considering real-time constraints and related computations in our proposed method.

V. Conclusions

In this letter, we have studied three main concerns in homogeneous multicore embedded systems, i.e., low power consumption, high reliability, and real-time computing. In order to overcome these concerns, we have proposed an RL-based DVFS method to reduce power consumption without reliability degradation for executing sporadic tasks. The proposed method has the ability to adapt to different situations using learning capabilities to prevail over the above-mentioned concerns. Experimental evaluation was done on different configurations and with different numbers of tasks to investigate the efficiency of the proposed method. Our experiments show that our proposed method works efficiently than other existing works for reducing power consumption without reliability degradation and deadline misses.

References

[1] A. Yeganeh-Khaksar, M. Ansari and A. Ejlali, "ReMap: Reliability Management of Peak-Power-Aware Real-Time Embedded Systems through Task Replication," in *IEEE Transactions on Emerging Topics in Computing*, doi: 10.1109/TETC.2020.3018902.

[2] 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," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 10, pp. 2338-2354, 1 Oct. 2019.

[3] W. Liu, Y. Tan and Q. Qiu, "Enhanced Q-learning algorithm for dynamic power management with performance constraint," 2010 *Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*, Dresden, 2010, pp. 602-605.

[4] Y. Tan, W. Liu and Q. Qiu, "Adaptive power management using reinforcement learning," 2009 *IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, San Jose, CA, 2009, pp. 461-467.

[5] 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* 15, 4, Article 33, December 2019.

[6] K. Swaminathan, N. Chandramoorthy, C. Cher, R. Bertran, A. Buyuktosunoglu and P. Bose, "BRAVO: Balanced Reliability-Aware Voltage Optimization," *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA 2017)*, Austin, TX, 2017, pp. 97-108.

[7] M. Ansari, S. Safari, A. Yeganeh-Khaksar, M. Salehi and A. Ejlali, "Peak Power Management to Meet Thermal Design Power in Fault-Tolerant Embedded Systems," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 1, pp. 161-173, 1 Jan. 2019.

[8] M. Ansari, A. Yeganeh-Khaksar, S. Safari and A. Ejlali, "Peak-Power-Aware Energy Management for Periodic Real-Time Applications," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 4, pp. 779-788, April 2020.

[9] S. Pagani, P. D. S. Manoj, A. Jantsch and J. Henkel, "Machine Learning for Power, Energy, and Thermal Management on Multicore Processors: A Survey," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 1, pp. 101-116, Jan. 2020.

[10] S. Yue, D. Zhu, Y. Wang, and M. Pedram, "Reinforcement learning based dynamic power management with a hybrid power supply," in *30th IEEE International Conference on Computer Design (ICCD)*, Sept 2012, pp. 81 – 86.

[11] Richard Ernest Bellman. 2003. *Dynamic Programming*. Dover Publications, Incorporated.

[12] M. Ansari, M. Salehi, S. Safari, A. Ejlali and M. Shafique, "Peak-Power-Aware Primary-Backup Technique for Efficient Fault-Tolerance in Multicore Embedded Systems," in *IEEE Access*, vol. 8, pp. 142843-142857, 2020.

[13] M.R. Guthaus, J.S. Ringenberg, D. Ernst, T.M. Austin, T. Mudge, and R.B. Brown, "MiBench: A Free, Commercially Representative Embedded Benchmark Suite," *Proc. Fourth IEEE Ann. Workshop on Workload Characterization*, pp. 3-14, 2001.

[14] Sheng Yang et al., "Adaptive energy minimization of embedded heterogeneous systems using regression-based learning," *2015 25th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Salvador, 2015, pp. 103-110.

[15] S. Manoj P. D., H. Yu and K. Wang, "3D Many-Core Microprocessor Power Management by Space-Time Multiplexing Based Demand-Supply Matching, " in *IEEE Transactions on Computers*, vol. 64, no. 11, pp. 3022-3036, 1 Nov. 2015.

[16] N. Binkert, et al., "The gem5 simulator," *ACM SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, May 2011.

[17] U. Gupta, S. K. Mandal, M. Mao, C. Chakrabarti and U. Y. Ogras, "A Deep Q-Learning Approach for Dynamic Management of Heterogeneous Processors," in *IEEE Computer Architecture Letters*, vol. 18, no. 1, pp. 14-17, 1 Jan.-June 2019.

[18] Z. Sun, H. Zhou and S. X. -. Tan, "Dynamic Reliability Management for Multi-Core Processor Based on Deep Reinforcement Learning," *2019 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, Lausanne, Switzerland, 2019, pp. 217-220.

[19] V. Rathore, V. Chaturvedi, A. K. Singh, T. Srikanthan and M. Shafique, "Life Guard: A Reinforcement Learning-Based Task Mapping Strategy for Performance-Centric Aging Management," *2019 56th ACM/IEEE Design Automation Conference (DAC)*, Las Vegas, NV, USA, 2019, pp. 1-6.