Received September 28, 2018, accepted November 6, 2018, date of publication November 22, 2018, date of current version December 27, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2882941

Contention & Energy-Aware Real-Time Task Mapping on NoC Based Heterogeneous MPSoCs

HAIDER ALI[1], UMAIR ULLAH TARIQ[2], YONGJUN ZHENG[1], XIAOJUN ZHAI[3], AND LU LIU[1]

[1]College of Engineering and Technology, University of Derby, Derby DE22 1GB, U.K.
[2]The School of Computer Science and Engineering, University of New South Wales, Sydney, NSW 2052, Australia
[3]School of Computer Science and Electronics Engineering, University of Essex, Colchester CO4 3SQ, U.K.

Corresponding author: Xiaojun Zhai (xzhai@essex.ac.uk)

ABSTRACT Network-on-Chip (NoC)-based multiprocessor system-on-chips (MPSoCs) are becoming the de-facto computing platform for computationally intensive real-time applications in the embedded systems due to their high performance, exceptional quality-of-service (QoS) and energy efficiency over superscalar uniprocessor architectures. Energy saving is important in the embedded system because it reduces the operating cost while prolongs lifetime and improves the reliability of the system. In this paper, contention-aware energy efficient static mapping using NoC-based heterogeneous MPSoC for real-time tasks with an individual deadline and precedence constraints is investigated. Unlike other schemes task ordering, mapping, and voltage assignment are performed in an integrated manner to minimize the processing energy while explicitly reduce contention between the communications and communication energy. Furthermore, both dynamic voltage and frequency scaling and dynamic power management are used for energy consumption optimization. The developed contention-aware integrated task mapping and voltage assignment (CITM-VA) static energy management scheme performs tasks ordering using earliest latest finish time first (ELFTF) strategy that assigns priorities to the tasks having shorter latest finish time (LFT) over the tasks with longer LFT. It remaps every task to a processor and/or discrete voltage level that reduces processing energy consumption. Similarly, the communication energy is minimized by assigning discrete voltage levels to the NoC links. Further, total energy efficiency is achieved by putting the processor into a low-power state when feasible. Moreover, this approach resolves the contention between communications that traverse the same link by allocating links to communications with higher priority. The results obtained through extensive simulations of real-world benchmarks demonstrate that CITM-VA approach outperforms state-of-the-art technique and achieves an average $\sim 30\%$ total energy improvement. Additionally, it maintains high QoS and robustness for real-time applications.

INDEX TERMS Contention-aware, DAG, real-time, real-world, task, mapping, scheduling, NoC, links, MPSoC, tiles, DVFS, DPM, CITM-VA, ELFTF, ECM, energy efficiency, QoS.

I. INTRODUCTION

Energy dissipation over the past decade in System-on-Chips (SoCs) has become a captious design constraint as it limits the performance, reliability and battery life [1]. Therefore, energy management techniques such as Dynamic Voltage and Frequency Scaling (DVFS) or Dynamic Power Management (DPM) are adopted to optimize the energy consumption. DVFS reduces concurrently the supplied voltage and frequency of the DVFS-enabled processor when performance requirement is low [1], [2]. DPM shutdowns the processor or switches it to a low-power state i.e. sleeping mode when it is in an idle state and similarly wakes it up when needed [3]. Modern embedded systems use SoCs which integrate processors, memory, advanced peripherals and power management circuitry on a single chip [4].

Multiprocessor System-on-Chips (MPSoCs) are widely deployed in high-performance computing and application-specific embedded systems such as gaming and aerospace for real-time response. Moreover, they offer energy efficiency and performance advantages over uniprocessor architectures [1], [5], [6]. Thus, MPSoCs are becoming the computing engines in embedded systems for real-time applications.

A dramatic increase in their use is expected in the upcoming years and there will be hundreds of processors on a single chip [5]. Subsequently, Network-on-Chip (NoC) will replace the traditional bus-based communication due to its limited band width, high latency and poor scalability [7].

Heterogeneous MPSoCs significantly reduce the energy consumption and dramatically enhance the performance compared to homogeneous MPSoCs [8]. They contain interconnected DVFS-enabled processors exhibiting different power-performance profiles and computing capabilities [9], [10]. Samsung Exynos 5422 is a popular example of heterogeneous MPSoC which is used in Galaxy S5 smart-phone. It contains of high-performance ARM Cortex-A15 processor and energy efficient ARM Cortex-A7 [11]–[13]. Few other commercial heterogeneous MPSoCs include Cell by IBM and Toshiba, Texas Instruments OMAP^{TM}1510, OMAP^{TM}3630, Apple A5X, and SHAPES a NoC based heterogeneous MPSoC [14].

Mapping is basically proper allocation of a set of tasks representing an application on the processors of MPSoC architecture in order to reduce either energy consumption or enhance the performance of the system [15]. Task mapping on MPSoCs is a well known NP-hard problem [16]. Therefore, an optimal solution does not exist subsequently, heuristics are deployed to obtain a near optimal solution [17]. Heuristics for task mapping approaches use different formulation based on Multi Integer Linear Programming (MILP) [18], Integer Linear Programming (ILP) [7], [19]–[21] and Non Linear Programming (NLP) [22]. Additionally searching based algorithms are also developed using Genetic Algorithm (GA) [23], [24], Ant Colony Optimization (ACO) [25], Particle Swarm Optimization (PSO) [19], [26] and Simulated Annealing (SA) [21], [27]. Among these algorithms, GA is a popular and widely adopted algorithm for task mapping on MPSoCs.

Dynamic and static are the two types of task mapping. In dynamic mapping, the tasks are assigned to the processors at runtime while in static mapping task set is allocated to the processors before an embedded system runs [28]. Dynamic mapping can fully utilize the available resources of the embedded system but increases the optimization complexity significantly [29]. Static mapping can be used in various real-time Internet-of-Things (IoT) based multimedia applications for example human gait analysis [30], remote ultrasound system [31], object or person tracking [32], surveillance [33], [34] and human recognition [35]. IoT paradigm has enabled embedded systems (smart nodes, sensors, actuators) to interconnect them to the Internet using networking technologies for physical and/or environmental conditions monitoring/control purposes as shown in FIGURE 1 [36]–[38]. Quality-of-Service (QoS) is an important concept for energy constrained embedded systems in IoT because degradation of QoS is unacceptable in health-care and safety related critical applications [39]–[41]. Therefore, it is important that QoS is addressed all the time along with energy savings in IoT.

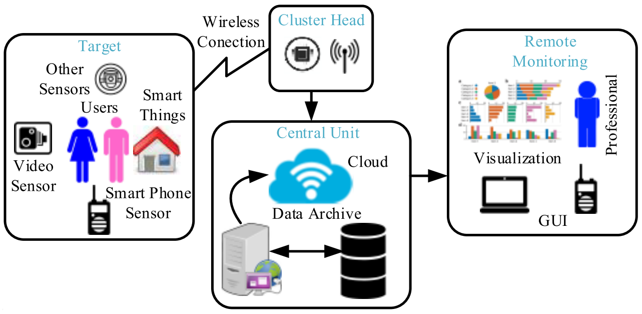

FIGURE 1. Data flow in IoT for real-time applications.

In this work, NoC based heterogeneous MPSoC architecture with DVFS-enabled processors is considered while contention and energy-aware static mapping for real-time Directed Acyclic Graph (DAG) tasks with individual deadlines and precedence constraints is studied.

First, task mapping, scheduling, and voltage scaling are performed in an integrated manner. Unlike other approaches which map the tasks first and then assigns voltage levels separately. Our Contention-aware Integrated Task Mapping and Voltage Assignment (CITM-VA) scheme guides the tasks and communications mapping to a more energy efficient solution. Moreover, both DVFS and DPM are integrated to reduce the total energy consumption.

Second, the proposed CITM-VA static energy management scheme saves communication energy by minimizing the communication over the NoC. It further reduces the communication energy by scaling the voltage levels of the NoC links and assigns communications voltage level such that communication energy is minimized. Hence the available slack is efficiently shared between the communications and tasks. Furthermore, contentions among concurrent tasks and communications are resolved by prioritizing the execution of high priority tasks and communications over low priority ones.

Third, our experimental results are generated from simulations conducted on five real-world benchmarks adopted from Embedded Systems Synthesis Benchmarks (E3S) [42]. The results are compared to state-of-the-art Energy-efficient Contention-aware Mapping (ECM) static energy management scheme developed by Li and Wu [24]. The proposed CITM-VA approach outperforms ECM in terms of energy savings and QoS. Compared to ECM, CITM-VA reduces the average total energy consumption by $\sim 30\%$.

The rest of the paper is organized as follows: Section II reviews existing task mapping and scheduling approaches using multiprocessors systems. Preliminaries are discussed in Section III. Section IV presents the proposed static contention and energy-aware scheme. The results examined in Section V, and Section VI concludes this paper.

II. RELATED WORK

Motivated by the fact that MPSoCs are high-performance green computing platform several studies have investigated the static task mapping and scheduling problem. Static task

Mapping and scheduling strategies targeting energy savings can be categorized into (A) computation (B) communication and (C) Total.

A. COMPUTATION ENERGY REDUCTION

Srinivasan and Chatha [27] developed SA based energy optimization approach called $(LPPWU)_{sa}$ for MPSoC platform with DVFS-enabled homogeneous processors using bus-based communication. The $(LPPWU)_{sa}$ reduced overall run time and combined DVFS and DPM for achieving maximum energy savings. This approach first only combines DVFS with the scheduling and after that separately deploys DPM at the final step to minimize the overall energy consumption.

Tosun [43] mapped periodic independent tasks on heterogeneous MPSoC architecture to reduce computation energy consumption. A heuristic algorithm using the Earliest Deadline First (EDF) strategy is deployed for task mapping and optimal voltage levels are assigned using DVFS. The investigation assumes independent task model and does not perform experiments on dependent tasks. Moreover, task duplication is used which can negatively affect the memory usage of the MPSoC platform.

Chen *et al.* [18] formulated the energy optimization problem as MILP using NoC based homogeneous MPSoC architecture for dependent real-time tasks represented by a DAG. A Non-preemptive schedule is generated and discrete voltage level is assigned to each task for energy consumption optimization. However, this study does not explicitly consider communication energy overhead i.e. interprocessor communications and heterogeneous MPSoC architecture for further energy savings.

An ILP-based meta-heuristic called Shuffled Frog Leaping Algorithm (SFLA) is proposed by Zhang *et al.* [19] to map real-time tasks on bus-based communication MPSoC platform consisting of heterogeneous processors. The study focuses only on real-time independent periodic applications while does not consider inter-task communication and task precedence constraints.

Tariq and Wu [22] investigated the problem of energy-aware scheduling of tasks with conditional precedence constraints on shared-memory homogeneous MPSoC. Their objective was to minimize the total computation energy. In their approach they first map the tasks to processors and then perform voltage scaling. They have proposed an NLP based algorithm that assigns optimal continuous voltage level to each task given and initial schedule. The major drawback of their approach is that they assume processors can operate at any voltage level between minimum and maximum voltage and frequency levels. This is a not a practical assumption as the processors in MPSoCs only operate on the discrete voltage and frequency levels.

B. COMMUNICATION ENERGY OPTIMIZATION

Shin and Kim [44] considered a NoC with voltage scalable links and proposed a GA based approach for reducing the communication energy by finding the optimal voltages for communications on the links. Similarly, Wang *et al.* [45] studied the static mapping approach deploying Adaptive Genetic Algorithm (AGA) for communication energy management. Chou and Marculescu [46] improved the contention of NoC architecture for MPSoCs platform using DAG. Mapping problem is formulated as an ILP for improving congestion control efficiency and providing best-effort communication in the network. A Linear Programming (LP) based heuristic is also developed to overcome the scalability issue. However, these studies have not considered the reduction of computation energy consumed by the processors of MPSoCs architecture.

Wang *et al.* minimized the inter-processor communication overhead and improved memory usage and throughput using traditional bus-based interconnect infrastructure homogeneous MPSoC platform. First, intra-data dependencies are transformed into inter-data dependencies in the DAG. Second, Heuristic Memory-Aware Task Scheduling (HMATS) is proposed for near optimal task schedule [45]. Since the scheduling heuristic algorithm focus only on traditional bus based communication, therefore cannot be extended to NoC based multiprocessor systems.

Sing *et al.* [47] proposed a contention-aware, energy efficient, duplication-based mixed integer programming (CEED-MIP) formulation in order to schedule task graphs on heterogeneous NoC-MPSoC architecture. This approach duplicates some tasks on the processor to reduce the inter-processor communication energy and avoid traffic congestion. The study fails to minimize processing energy consumption. Furthermore, duplication of tasks adversely affects overall systems energy savings which are not considered in the energy model.

C. TOTAL ENERGY SAVINGS

Wang *et al.* [20] optimized communication and computation energy for real-time streaming DAG task using bus communication based homogeneous MPSoC architecture. Task mapping and scheduling problem is formulated as ILP and the schedule length is minimized by reducing the inter-processor communication overhead. In another study, Wang *et al.* deployed homogeneous MPSoC architecture for real-time streaming applications and integrated task level coarse-grained software pipelining with DPM and DVFS to reduce the total energy consumption. A two-phase energy optimization algorithm is developed wherein the first phase DAG is transformed into independent task model using re-timing. In the second phase GA based algorithm known as GeneS is used to find a feasible schedule and DVFS and DPM are used to reduce computation and inter-processor communication energy consumption [23]. Though, processing and communication energies are minimized but both the studies do not consider NoC based communication. Furthermore, in this study all processors are assumed to be homogeneous.

Huang *et al.* [21] extended the ILP formulation to both communication and processing energies optimization of NoC-MPSoCs platform with heterogeneous processors.

Li and Wu [24] proposed a two-step contention and energy-aware real-time task mapping on NoC based homogeneous MPSoC architecture. First, task mapping is formulated as a quadratic binary programming problem that minimizes the communication energy. Second voltage levels are assigned to each task and communication using GA.

Tariq *et al.* [7] used NoC based heterogeneous MPSoC and minimized the total energy consumption of tasks with conditional precedence constraints. They have proposed an Iterative Offline Energy-aware Task and Communication Scheduling (IOETCS) Algorithm that collectively performs scheduling and voltage scaling. They have proposed an NLP based algorithm that given an initial schedule generated by Earliest Successor-Tree-Consistent Deadline First algorithm assigns each task and communication optimal voltage and frequency levels within a continuous voltage and frequency range. The optimal continuous voltage and frequency levels are then mapped to valid discrete voltage and frequency levels using either an ILP or Heuristic based algorithms. However, in their approach they have not integrated DPM with DVFS and assume that no energy is consumed during the idle time slots.

Unlike the aforementioned studies we consider NoC based heterogeneous MPSoC architecture with distributed memory for real-time dependent task represented by a DAG. Moreover, We reduce both processing and communication energy consumption and integrate DVFS and DPM in our developed CITM-VA heuristic while explicitly considering NoC links contention. To the best of our knowledge, we have proposed the first approach to solve this problem.

III. PRLIMINARIES

In this section we introduce models on which the proposed contention and energy-aware static task mapping approach CITM-VA is based. In this paper tiles and processors are used interchangeably.

A. APPLICATION MODEL

A real-time application with dependent tasks can be modeled by a *Directed Acyclic Graph (DAG)* as shown in FIGURE 2. A DAG, $G(V, E, X)$ is an edge weighted task graph. $V = \{v_1, v_2, \ldots, v_n\}$ is a set of nodes and each node $v_i \in V$ represents a task (a sequential chunk of execution). Each task v_i has an execution requirement in worst case clock cycles on a processor pe_j represented by $NCC_{(i,j)}$ as well as an implicit deadline. $E \subseteq V \times V$ denotes a set of edges where each edge $(v_i, v_j) \in E$ represent a data dependency relation between tasks v_i and v_j. X is a set of edge weights. Thus, $\chi_{(i,j)}$ the edge weight of an edge (v_i, v_j) is the volume of data sent from v_i to v_j in units of bits. Each task in DAG has an individual deadline.

B. PLATFORM MODEL

NoC based MPSoC consisting k number of heterogeneous tiles is considered as demonstrated in FIGURE 3(a). Each tile is comprised of local memory, processor (pe) and a network interface. Therefore, MPSoC contains a set $P = \{pe_1, pe_2, \ldots, pe_k\}$ of k DVFS-enabled processors. The links that provide connection between the router (represented by R) and tile are termed as *Local Links* while *Global* links interconnect the routers with each for data communication purposes. Each processor $pe_i \in P$ can operate at $\{(V_{dd_1}, f_1), (V_{dd_2}, f_2), \ldots, (V_{dd_n}, f_{n_i})\}$ set of n_i discrete voltage-frequency levels. Moreover, each processor supports DPM and can be switched into different power modes.

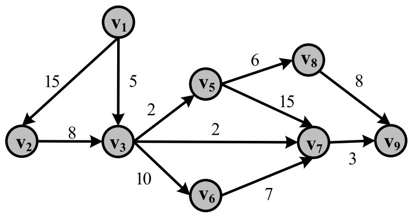

FIGURE 2. DAG representation of real-time application.

C. COMMUNICATION INTERCONNECT MODEL

A 2D mesh topology NoC architecture is assumed for inter-processor communication. It consists N_x rows and N_y columns of the routers therefore, k number of routers are equal to $N_x \times N_y$. Each routers is comprised of five ports to communicate with neighbor routers and a processor as shown in FIGURE 3(b). Each ports has a Link and buffer. All links are identical and full duplex with band width b_w. Similar to the processors the links in NoC can operate at n set of discrete voltage/frequency levels i.e. $\{(V_{dd_1}, f_1), (V_{dd_2}, f_2), \ldots, (V_{dd_n}, f_n)\}$.

We assume a simple and energy efficient Wormhole (WH) packet switching technique for NoC communication. WH splits the data packet into small pieces called FLITS and they are delivered in a pipelined fashion in the network. Furthermore, we assume widely used deterministic XY routing scheme. The distance between two processors pe_i and pe_j in 2D mesh is given by the Manhattan distance $\eta_{i,j} = |x_i - x_j| + |y_i - y_j|$, where (x_i, y_i) are the coordinates of processor pe_i and (x_j, y_j) are the coordinates of processor pe_j in the mesh.

D. ENERGY MODEL

We consider the energy consumed by processors, routers and network links in our energy model. The dynamic power P_{d_i} dissipated in executing a task v_i on a pe_j is given by the following equation [22], [48]:

$$P_{d_i} = C_{eff_i} V_{dd_j}^2 f_j \tag{1}$$

where, V_{dd_j}, f_j, and C_{eff_i} denote the supply voltage, operating frequency and the effective load switching capacitance, respectively. This mathematical relation shows that decrease in dynamic power occurs when the supplied voltage is reduced thus, Equation (1) serves as the baseline for DVFS.

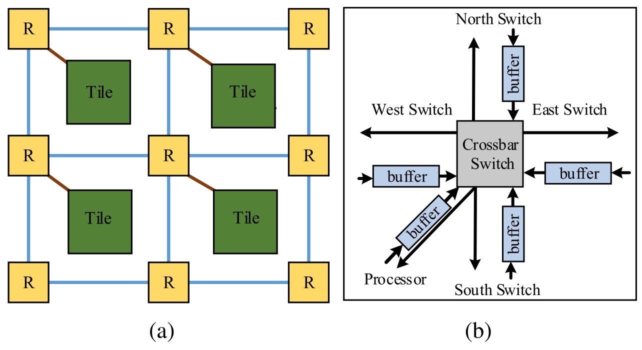

FIGURE 3. 2D-mesh topology based NoC-MPSoC. (a) NoC-MPSoC architecture. (b) NoC router structure.

The total power (P_{T_i}) dissipated in executing a task on a pe_j is the sum of the dynamic power, the static power and inherent power P_{on} required to keep the processor on i.e. idle power when no task is running on the processor. At V_{dd_j} and f_j, the P_{T_i} is calculated as follows [22]:

$$P_{T_i} = P_{d_i} + L_g(V_{dd_j}K_3 e^{K_4 V_{dd_j}} e^{K_5 V_{bs}} + |V_{bs}|I_{j_n}) + P_{on} \tag{2}$$

where K_3, K_4 and K_5 are technology dependent constants, L_g is the number of logic gates in the circuit and I_j and V_{bs} are junction leakage current and body-bias voltage, respectively.

As $E = P \times t$, where t shows time so, the energy E_i consumed in executing a task v_i on p_j is given as follows [7], [18]:

$$E_i = C_{eff} V_{dd_i}^2 NCC_i + L_g(V_{dd} K_3 e^{K_4 V_{dd}} e^{K_5 V_{bs}} + |V_{bs}|I_j)t_i \tag{3}$$

where t_i is the execution time of v_i and is given by $t_i = \frac{NCC_{(i,j)}}{f_j}$. The operating frequency f and supply voltage V_{dd} are related by the following equation [48]:

$$f = (V_{dd} + V_{th})^{\alpha} / K_6 L_d V_{dd} \tag{4}$$

where V_{th} is the threshold voltage, K_6 is the process dependent constant, L_d is the logic depth of the processor critical path and α reflects velocity saturation ($1.4 \leq \alpha \leq 2$). So, according to Equation (4) energy consumption rely on the voltage and frequency level assigned to the tasks.

The energy consumed by the processor when it is idle in an active mode is given as follows:

$$E_{idle} = P_a \times t_{idle} \tag{5}$$

where P_a is the power consumed by the processor in active mode and t_{idle} is the time period for which the processor is idle. Similarly, the energy consumed by a processor in the sleep mode is calculated by:

$$E_{sleep} = P_{sleep} \times t_{sleep} \tag{6}$$

where P_{sleep} is the power dissipated by the processor in sleep mode and t_{sleep} is the duration for which processor stays in sleep mode. Since $P_a > P_{sleep}$, energy efficiency can be achieved by switching the processor into a sleep mode. However, there are switching costs associated in transitioning the processor between active and sleep modes. The processor break even time t_{BET} represents the shortest duration of idle time interval that justifies processor's transition from active mode to sleep mode. Thus, if this interval is shorter than t_{BET} the mode switch overheads are larger than the energy saving and therefore, transition to low power mode should be avoided. The definition of t_{BET} is given as follows:

$$t_{BET} = \max\left\{t_{sw}, \frac{E_{sw}}{P_a - P_{sleep}}\right\} \tag{7}$$

where t_{sw} and E_{sw} are total switching time interval and total switching energy overhead respectively.

Under the fixed operational frequency f_l of the links the time taken to transmit the message ($\chi_{i,j}$) of a communication is in general dominated by the serialization delay. The execution time $t_{i_{com}}$ of transmitting a message for communication $e_i = (v_s, v_d)$ is given as follows [7]:

$$t_{i_{com}} = \frac{\chi_{(i,j)}}{f_l \times b_w} \tag{8}$$

Let the parent node v_s of e_i be mapped on pe_i and its child node v_k be mapped on pe_j. Then the hop count between pe_i and pe_j is given as $\eta_{i,j}$. The energy consumed in transmitting one bit of a message e_i is $E_{bit} = E_{Rbit}(\eta_{(i,j)} + 1) + \eta_{(i,j)}E_{lbit}$, where E_{Rbit} is the energy consumed by a router in transmitting one bit and E_{lbit} is the energy consumed by links in transmitting a bit. The energy E_{c_i} consumed in transmitting $\chi_{(i,j)}$ volume of data is calculated as follows [7], [24]:

$$E_{c_i} = \chi_{(i,j)}E_{Rbit}(\eta_{(s,d)} + 1) + \chi_{(i,j)}E_{lbit}\eta_{(s,d)} \tag{9}$$

$$E_{lbit} = \frac{P_i}{f_j \times b_w} \tag{10}$$

where P_i shows the total power consumption for one bit on the links that e_i traverses at f_i. Thus, P_i is the sum of static power (P_s) and dynamic power (P_d) i.e. $P_i = P_{d_i} + P_{s_i}$ [48].

Inserting Equation (10) in Equation (9) yields the following equation:

$$E_{c_i} = \chi_{(i,j)}((\eta_{(s,d)} + 1)E_{R_{bit}} + \eta_{(s,d)}\frac{P_i}{f_j \times b_w}) \tag{11}$$

Equation (11) indicates that communication energy can be reduced by assigning optimal discrete frequency or voltage levels (as voltage and frequency are interchangeable according to Equation (4)) to mesh topology NoC links for transmitting the data.

IV. CONTENTION AND ENERGY-AWARE APPROACH

Energy optimization of real-time tasks with precedence and deadline constraints count on the mapping because heterogeneous MPSoC architecture consists of different performance profiles processors. Moreover, energy saving is associated with the order in which both the tasks and communications are executed. Thus, a significant amount of energy reduction can be achieved by prioritizing shorter deadlines tasks and communications among the nodes. This

TABLE 1. Terms and notations.

Term/ Notation	Definition
κ	Total number of mapping solutions to the problem.
Π	A matrix of κ mapping. Each row represents processor to which task v_i is mapped, $1 \le i \le \lvert V\rvert$.
Ψ	A matrix of κ voltage assignments. Each row represents the voltage level of task or communication node v_i, $1 \le i \le \lvert V\rvert\lvert E\rvert$.
$\Pi[\eta][i]$	Processor where task v_i is mapped for η mapping.
$\Psi[\eta][i]$	Voltage level assigned to task v_i for η voltage assignment.
$rand()$	Returns a uniform random number in interval $(0, 1)$.
Ω	Maximum number of iterations (Pre-defined).
$Stagnation$	No improvement achieved in energy reduction for a predefined number of iterations
$Solution$	A row from Π and the same row from Ψ is together is a solution.
G'_e	A copy of the extended graph G_e
$cSet(v_i)$	Set of concurrent nodes to v_i
$ISuc(v_i)$	Set of immediate successors of v_i.
PRI	It shows priority of the tasks
$NULL$	Processor is idle

priorities assignment strategy enables DVFS technique to efficiently utilize the available slack by assigning lower voltage levels to the tasks. Moreover, tasks and communication deadlines of real-time application are not violated by DVFS technique which ensures high QoS. Furthermore, DPM also plays a vital role when processor's leakage power in idle mode is taken into consideration and energy efficiency can be increased by switching the processors into a low-power state. Therefore, the quality and energy efficiency of an approach targeting heterogeneous MPSoCs architecture is influenced by four factors: (1) Task Mapping (2) Voltage Scaling (3) Task Ordering and (4) Slack Power Management. Therefore, algorithms 1-4 are developed for these steps implemented in an integrated manner. Moreover, the notations and terms used in these algorithms are listed in TABLE 1.

Before CITM-VA approach is explained which is demonstrated in Algorithm 1, extended graph (G_e) is defined. Basically G_e is the transformed version of a traditional DAG (G). So, G_e is generated by inserting additional node V_s in G for each edge (v_i, v_j) whose head node V_j and tail node V_i are mapped to different processors. Two edges (v_i, v_s) and (v_s, v_j) are put in place of the edge (v_i, v_j) in G_e. These additional nodes represent communication nodes. G_e can be denoted as $G(V + V^*, E)$, where V shows set of task nodes, V^* indicates set of communication nodes and E is edges set. The transformed graph G_e of the traditional G consisting of nine nodes given in FIGURE 2 is exhibited in FIGURE 4.

A. CITM-VA

Algorithm 1 shows CITM-VA approach that fundamentally consists of four implementation steps i.e. (1) Initial Solution (2) Solution Refinement (3) Stagnation Control, and (4) Termination. Each step of the CITM-VA approach is explained as follows:

Algorithm 1 CITM-VA

```
input : A DAG $G$, tasks Deadlines, an MPSoC, total number of
        iterations $\Omega$ and total number of initial solutions $\kappa$
output: Task to processor mapping $map$ and voltage levels $vol$
        corresponding to tasks an communications.
1  Construct two matrices $\Pi$ and $\Psi$ of zeros having dimensions
   $\kappa \times |V|$ and $\kappa \times |V||E|$ respectively;
2  for $\eta \leftarrow 1$ to $\kappa$ do
3      for each $v_i \in G'_e$ do
4          if $v_i$ is task node then
5              $\Pi[\eta][i] \leftarrow \lceil rand()(|P|-1)+1 \rceil$;
6              $\Psi[\eta][i] \leftarrow$ Maximum Voltage Level;
7          else
8              $\Psi[\eta][i] \leftarrow$ Maximum Link Voltage Level;
9  for $\eta \leftarrow 1$ to $\kappa$ do
10     Construct an extended graph $G_e$ given a mapping ;
11     $[f, e] \leftarrow ELFTF(G_e, \Pi, \Psi, \eta)$;
12     Compute solution $fitness(f, e)$ value according to equation
       (12) ;
13 while 1 do
14     Sort solutions in descending order of their fitness values;
15     if stopping criteria satisfied then
16         if the solution with highest fitness feasible then
17             Copy first row of $\Pi$ to map and $\Psi$ to $vol$;
18         else
19             No feasible solution found;
20         break;
21     if stagnation detected then
22         Sort solutions in descending order of their fitness values;
23         Delete $\lfloor \frac{1}{2}\kappa \rfloor$ solutions with smaller fitness value;
24         Construct two matrices $\Pi'$ and $\Psi'$ of zeros having
           dimensions $\left\lfloor \frac{1}{2}\kappa \right\rfloor \times |V|$ and $\left\lfloor \frac{1}{2}\kappa \right\rfloor \times |V||E|$;
25         for $j \leftarrow 1$ to $\left\lfloor \frac{1}{2}\kappa \right\rfloor$ do
26             $\phi \leftarrow rand()$;
27             Copy $j^{th}$ row of $\Pi$ and $\Psi$ to $j^{th}$ row of $\Pi'$ and $\Psi'$
               respectively;
28             Randomly re-map $\lfloor \phi |V| \rfloor$ tasks in $j^{th}$ row of $\Pi'$ to
               other processors at maximum voltage level;
29             Construct the extended graph $G_e$ for mapping given
               by $j^{th}$ row of $\Pi'$;
30             $[f, e] \leftarrow ELFTF(G_e, \Pi', \Psi', j)$;
31             Compute the fitness $fitness(f, e)$ of the $j^{th}$ solution;
32             Merge the matrix $\Pi'$ with $\Pi$ and $\Psi'$ with $\Psi$;
33     Delete $\left\lfloor \frac{1}{3}\kappa \right\rfloor$ solutions with smaller fitness values;
34     Construct two matrices $\Pi'$ and $\Psi'$ of zeros having dimensions
       $\left\lfloor \frac{1}{3}\kappa \right\rfloor \times |V|$ and $\left\lfloor \frac{1}{3}\kappa \right\rfloor \times |V||E|$;
35     for $j \leftarrow 1$ to $\left\lfloor \frac{1}{3}\kappa \right\rfloor$ do
36         $\eta \leftarrow \lceil rand()(\kappa - \left\lfloor \frac{1}{3}\kappa \right\rfloor - 1) + 1 \rceil$;
37         Copy the $\eta$ row of $\Pi$ and $\Psi$ to $j^{th}$ row of $\Pi'$ and $\Psi'$
           respectively;
38         Construct the extended graph $G_e$ for mapping given by $j^{th}$
           row of $\Pi'$;
39         $[f, e] \leftarrow ReMap(G_e, \Pi', \Psi', j)$;
40         Compute the fitness $fitness(f, e)$ of the $j^{th}$ solution;
41     Merge the matrix $\Pi'$ with $\Pi$ and $\Psi'$ with $\Psi$;
```

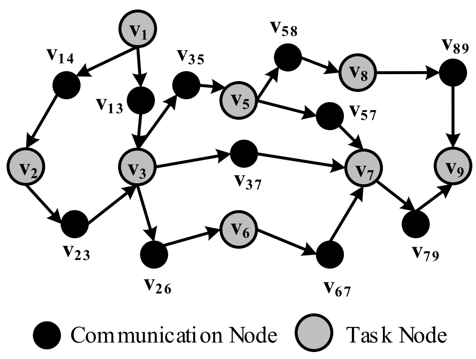

FIGURE 4. Extended DAG.

1) INITIAL SOLUTION

In the initial solution step of the CITM-VA first, two matrices Π and Ψ of $\kappa \times |V|$ are generated of $\kappa \times |V||E|$ dimensions respectively. Where κ denotes the input parameter of CITM-VA algorithm. It performs the following step to generate an initial solution.

1) First, task node v_i is randomly mapped to a processor, $1 \leq i \leq |V|$ (Line 5) to generate each row of matrix Π. Then each row of matrix Ψ is generated through maximum processor voltage assignment to a task v_i where it is mapped and maximum link voltage is assigned if v_i is a communication node $1 \leq i \leq |V||E|$ (Lines 6-8).
2) Now each row of Π and Ψ forms a solution. Therefore, to compute the fitness value, first a schedule is generated using Earliest Latest Finish Time First (ELFTF) algorithm described in Section IV-C (Lines 9-12).
3) Then ELFTF returns the feasibility and energy of each solution. Thereby, provided the energy and feasibility of each solution the fitness value is computed using fitness function given in Equation (12).

$$Fitness(feasible, e) = \begin{cases} \frac{1}{e} & \text{if feasible is true} \\ -\infty & \text{otherwise} \end{cases} \quad (12)$$

2) SOLUTION REFINEMENT

The obtained solutions from step 1 are sorted in a decreasing order of the fitness value in each iteration of the while loop (Lines 13-41).

1) First, $\lfloor \frac{1}{3}\kappa \rfloor$ solutions are deleted with smaller fitness value. Then $\lfloor \frac{1}{3}\kappa \rfloor$ new solutions are generated from the remaining solutions with higher fitness values (Lines 33-41).
2) Second, a new solution from an existing solutions is generated by re-mapping a task node to a processor and/or voltage level or a communication to a voltage level such that total energy is minimized and deadlines are satisfied by using *ReMap* algorithm which is explained in Section IV-B (Line 39).
3) Finally, the solution is updated if energy savings occur otherwise the existing solutions remain unchanged using the *ReMap* algorithm. The fitness value of each solution is calculated and new solutions are combined with existing solutions to generate κ solutions for the next iteration.

3) STAGNATION CONTROL

As CITM-VA adopts an elitism based searching approach and keeps the best and discards the worst solution in each iteration. Therefore, CITM-VA is prone to stagnation similar to other elitist approaches. Stagnation occurs when no improvement in the energy reduction is observed over a certain number of predefined iterations. In case of stagnation detection, a stagnation control strategy is followed to escape from local optimum (Lines 21-32).

1) We delete $\lfloor \frac{1}{2}\kappa \rfloor$ solutions with the smaller fitness values are deleted from κ solutions (Line 23).
2) Then, $\lfloor \frac{1}{2}\kappa \rfloor$ new solutions are generated from the remaining $\lfloor \frac{1}{2}\kappa \rfloor$ solutions (Lines 26-29).
3) After new solutions generation the fitness value for each existing and new solution is computed which collectively form κ solutions (Lines 30-32).

4) TERMINATION

CITM-VA is terminated when a maximum number of iteration Ω reach. Before termination CITM-VA performs a final check and determines the feasibility of the solution generated (Lines 16-20).

1) If the solution found with maximum fitness values is feasible then perform the operations and produce results of the given problem.
2) Otherwise CITM-VA cannot produce results for the given problem.

B. REMAP

Algorithm 2 shows our *ReMap* strategy that is used to map all the task nodes of G_e to the processors of MPSoC and communication nodes to NoC links respectively for higher energy savings.

1) First, task nodes are re-mapped to each processor and voltage/frequency level of the MPSoC architecture for a given problem (Lines 6-20).
2) Second, tentatively all the possible link voltages are re-assigned to each of the communication node (Lines 22-34).
3) After each re-map process, *ELFTF* is used to determine the energy consumption and feasibility of the schedule and the $Rank(v_i)$ of the newly generated schedule is calculated using Equation (13) (Line 16 and/or 29):

$$Rank(v_i) = \begin{cases} \dfrac{e - E^{new}}{\frac{NC(i,k)}{f_i^{new}} - \frac{NC(i,k)}{f_i^{cur}}} & \text{if } v_i \text{ is task node} \\ \dfrac{e - E^{new}}{\frac{x_i}{b_w f_i^{new}} - \frac{x_i}{b_w f_i^{cur}}} & \text{otherwise} \end{cases} \quad (13)$$

4) Finally, a node with highest rank is selected to update the solution.

Algorithm 2 ReMap

```
input  : A DAG $G_e$, matrix $\Pi$, matrix $\Psi$ and index $\eta$.
output: Schedule feasibility indicated by $f$, energy $e$.
1  $Rank^{select} \leftarrow -\infty; \tau \leftarrow -1; \rho \leftarrow -1; \Gamma \leftarrow -1; e \leftarrow 0;$
   $f \leftarrow true;$
2  for each $v_j \in G_e$ do
3      $Rank(v_i) \leftarrow -\infty;$
4      if $v_i$ is a task node then
5          for each $pe_k \in P$ do
6              Find the highest voltage level $V_{dd}^L$ on $pe_k$ that
               minimize total energy consumption;
7              $\Upsilon \leftarrow \{V_{dd}^{min}, \ldots, V_{dd}^L\}$;    ▷ $V_{dd}^{min}$: lowest voltage,
               voltage levels higher than $V_{dd}^L$ not considered
8              for each $V_{dd_s} \in \Upsilon$ do
9                  $tempProc \leftarrow \Pi[\eta][i]; \Pi[\eta][i] \leftarrow k;$    ▷
                   Remap task to processor $pe_k$
10                 Update $G_e$ according to new mapping;
11                 $tempVol \leftarrow \Psi[\eta][i]; \Psi[\eta][i] \leftarrow s;$
12                 $[Feasible, E^{new}] \leftarrow ELFTF(G_e, \Pi, \Psi, \eta);$
13                 $\Psi[\eta][i] \leftarrow tempVol; \Pi[\eta][i] \leftarrow tempProc;$
14                 if Feasible then
15                     Compute frequencies $f_i^{cur}$ and $f_i^{new}$
                       corresponding to voltages $\psi[\eta][i]$ and
                       $V_{dd_s}$ respectively;
16                     $Rank(v_i) \leftarrow \frac{E^{cur} - E^{new}}{\frac{NC(i,\Pi[\eta][i])}{f_i^{new}} - \frac{NC(i,\Pi[\eta][i])}{f_i^{cur}}};$
17                     if $Rank(v_i) > Rank^{select}$ then
18                         $Rank^{select} \leftarrow Rank(v_i); \tau \leftarrow i; \rho \leftarrow$
                           $s; e \leftarrow E^{new}; \Gamma \leftarrow k;$
19                 else
20                     break;
21     else
22         Find the highest link voltage level $V_{dd}^L$ that minimize
           total energy consumption; $\Upsilon \leftarrow \{V_{dd}^{min}, \ldots, V_{dd}^L\}$;
23         for each $V_{dd_s} \in \Upsilon$ do
24             $tempVol \leftarrow \Psi[\eta][i]; \Psi[\eta][i] \leftarrow s;$
25             $[Feasible, E^{new}] \leftarrow ELFTF(G_e, \Pi, \Psi, \eta);$
26             $\Psi[\eta][i] \leftarrow tempVol;$
27             if Feasible then
28                 Compute frequencies $f_i^{cur}$ and $f_i^{new}$
                   corresponding to voltages $\psi[\eta][i]$ and $V_{dd_s}$
                   respectively;
29                 $Rank(v_i) \leftarrow \frac{E^{cur} - E^{new}}{\frac{x_i}{b_w f_i^{new}} - \frac{x_i}{b_w f_i^{cur}}};$
30                 if $Rank(v_i) > Rank^{select}$ then
31                     $Rank^{select} \leftarrow Rank(v_i); \tau \leftarrow i; \rho \leftarrow s;$
                       $e \leftarrow E^{new};$
32             else
33                 break;
34 if $\tau \neq= -1$ then
35     if $v_i$ is a task node where $i = \tau$ then
36         $\Pi[\eta][\tau] \leftarrow \Gamma; \Psi[\eta][\tau] \leftarrow \rho;$
37     else
38         $\Psi[\eta][\tau] \leftarrow \rho;$
39 else
40     $f \leftarrow false;$
```

Algorithm 3 ELFTF

```
input  : An extended DAG $G_e$, matrix $\Pi$, matrix $\Psi$
         and current chromosome index $\eta$
output: Schedule feasibility indicated by binary
         variable $Feasible$, total energy $e_T$.
1  $Feasible \leftarrow True; e_T \leftarrow 0;$
2  for each $v_i$ in reverse topological order of $G_e$ do
3      if $v_i$ is a task node then
4          for each $v_j \in ISuc(v_i)$ do
5              Compute frequency $f_j$ of node $v_j$
               corresponding to voltage level $\Psi[\eta][j]$;
6              if $v_j$ is a task node then
7                  $LFT_i \leftarrow \min\{d_i, LFT_j - \frac{NCC(j,\Pi[\eta][j])}{f_j}\};$
8              else
9                  $LFT_i \leftarrow \min\{d_i, LFT_j - \frac{x_j}{f_j b_w}\};$
10     else
11         Compute frequency $f_j$ of task $v_j$ corresponding
           to to voltage level $\Psi[\eta][j]$;
12         $LFT_i \leftarrow LFT_j - \frac{x_j}{f_j b_w};$
13 Create a copy $G_e'$ of $G_e$;
14 $\forall v_i \in G_e$ $est_i \leftarrow 0$; ▷ Set the earliest start time (est) of
   all the tasks and communication nodes to zero
15 Insert all the source nodes in $G_e'$ to ready queue $R$;
16 while $R$ is not empty do
17     for each $v_i \in R$ do
18         $PRI_i \leftarrow est_i + LFT_i;$
19     Select the node $v_i$ with the smallest value of $PRI$
       from $R$;
20     if $v_i$ is a task node then
21         Find the latest scheduled task on $pe_i$ the
           processor where $v_i$ is mapped, $v_l$ is null if $v_i$ is
           the first task to be scheduled on $pe_i$ ;
22         $e_T \leftarrow$
           $e_T + E(i, \Psi[\eta][i], \Pi[\eta]][i]) + SPM(v_l, v_i)$ ;
23         $t_i^{start} \leftarrow est_i;$
24         $t_i^{finish} \leftarrow t_i^{start} + \frac{NC(i,\Pi[\eta][i])}{f_i};$
25         for each unscheduled node
           $v_j \in cSet(v_i) \cup ISuc(v_i)$ do
26             $est_j \leftarrow \max\{est_j, t_i^{finish}\};$
27     else
28         $e_T \leftarrow e_T + E_c(i, \Psi[\eta][i]); t_i^{start} \leftarrow est_i;$
29         $t_i^{finish} \leftarrow t_i^{start} + \frac{vol_i}{l_w f_i};$
30         for each unscheduled node
           $v_j \in cSet(v_i) \cup ISuc(v_i)$ do
31             $est_j \leftarrow \max\{est_j, t_i^{finish}\};$
32     if $t_i^{finish} > d_i$ then
33         $Feasible \leftarrow False;$
34     Delete $v_i$ and all its outgoing edges from $G_e'$;
35     Insert all the source nodes in $G_e$ to $R$;
```

Algorithm 4 SPM

```
input  : Nodes $v_i$ and $v_j$
output: Energy $E$ consumed during the idle interval.
1 $t_{idle} \leftarrow \rho_j - \zeta_i$;                ▷ If $v_i$ is NULL then $\zeta_i \leftarrow 0$
2 Calculate $t_{BET}$ using equation (7);
3 if $t_{idle} \geq t_{BET}$ then
4     $E \leftarrow (t_{idle} - t_{sw})P_{sleep} + E_{sw}$;
5 else
6     return $E \leftarrow E_{idle}$;
```

C. ELFTF

CITM-VA algorithm examines the feasibility of the schedule under given mapping and voltage assignment (for tasks and communications) by calling Algorithm 3. Algorithm 3 demonstrates *ELFTF* approach. The two major steps performed by *ELFTF* are explained as follows:

1) For each task and communication node first the latest finish time is computed (Lines 2-12).
2) A schedule for nodes is generated based on their priorities represented by *PRI* where *PRI* is equal to the sum of latest finish time and the earliest start time (16-35). Nodes with smaller value of *PRI* have higher priority over nodes with larger value of *PRI*. The nodes with higher priorities are scheduled earlier in time compared to nodes with lower priorities.

To achieve the goal of the second step first a set named $cSet(v_i)$ is defined. Before $cSet(v_i)$ is defined concurrent nodes are defined. Two nodes in G_e are concurrent if they are not reachable from each other in G_e. For a task node v_i, $cSet(v_i)$ is defined as a set of task nodes concurrent to v_i and mapped on the same processor where v_i is mapped. For a communication node v_i, $cSet(v_i)$ is a set of communication nodes concurrent to v_i and have conflict with v_i (communication nodes have conflicts if they use same NoC links). Therefore, after a node is scheduled the earliest start time of all the nodes that belong to $cSet(v_i)$ and $ImSuc(v_i)$ are updated to finish time of v_i. This ensures that no two nodes that are concurrent are scheduled at the same time and the nodes with smaller priority are scheduled later than nodes with higher priority. The schedule feasibility is determined and its energy is also computed in the process.

D. SLACK POWER MANAGEMENT

There may be slack available after voltage scaling. As discussed in section III-D the processor can remain in active state during the idle period or it can switch to sleep state. This is determined by Algorithm 4 the Slack Power Management (SPM). After scheduling a node we call SPM (in ELFTF, Line 22). It performs the following steps:

1) We first determine the length of idle time slot t_{idle} (Line 1).
2) We calculate break even time t_{BET} (Line 2).
3) If idle interval is greater than or equal to break even time then the processor is switched to sleep mode otherwise the processors stays in active mode (Lines 3-6).

TABLE 2. Real-world benchmarks description.

Real-world benchmark	No.of Tasks
MP3-encoder	16
Consumer	7
Networking	4
Office-automation	5
Auto-industry	9

V. RESULTS AND DISCUSSION

The energy performance of CITM-VA scheme is evaluated by comparing it to a GA based Energy-efficient Contention-aware Mapping (ECM) approach developed by Li and Wu [24]. Primarily ECM maps DAG tasks on NoC based homogeneous MPSoC architecture using quadratic binary programming to minimize the communication energy by applying relaxation-based iterative rounding. Then a GA is used to assign discrete voltages to the task and communication nodes for processing energy consumption reduction considering the deadline. It performs mapping and voltage scaling separately and does not consider LTF of the tasks to generate a suitable task schedule. Moreover, energy performance profiles of the processors are not considered for higher energy efficiency. Additionally energy consumption during the idle period of the processors is assumed negligible which is not a practical scenario. A certain amount of energy consumes to keep the processor on even if there is no task running on it.

A. EXPERIMENTAL SETUP

Five real-world benchmarks listed in TABLE 2 are selected from E3S [42], which is the most suited and widely adopted benchmarks suite in the automated system level allocation and scheduling research [42], [49]. In the experimental analysis, a NoC based MPSoC architecture with heterogeneous DVFS-enabled processors is used. Furthermore, the processors have the capability to switch into different power modes. The results are generated considering different scenarios and parameters. A 70 nanometers (nm) processor technology is adopted from [18] and the values of the different parameters are given in TABLE 3. These parameters show CMOS fabrication in 70 nm technology. The reason behind considering 70 nm CMOS technology is that leakage power consumption constitutes a significant portion of the total power [18], [48]. Subsequently, which makes it a suitable technology for CITM-VA which targets to optimize the leakage power consumption during the idle period of the processor for real-time applications. In the future dynamic power would be dominated by leakage power consumption in the modern embedded systems [50]. Therefore, static energy consumption optimization is important.

TABLE 3. 70 nm processor technology parameters.

Parameter	Value	Parameter	Value
K_1	0.063	K_2	0.153
K_3	5.38×10^{-38}	K_4	1.83
K_5	4.19	K_6	5.26×10^{-12}
C_{eff}	4.30×10^{-10}	α	2.00
I_j	4.80×10^{-10}	L_g	4.00×10^{6}
V_{bs}	- 0.70	V_{th}	0.244

TABLE 4. Processors power consumption modes.

Type 1		Type 2	
Parameters	Value	Parameters	Value
P_a	276 mW	P_a	555 mW
P_{sleep}	80.0 μ W	P_{sleep}	180 μW
P_{sw}	385 μ W	P_{sw}	483 μ w

TABLE 5. Type 1 processor operating speed.

Parameters	Value				
$V_{dd}(V)$	0.85	0.8	0.75	0.7	0.65
$f(GHz)$	2.10	1.81	1.53	1.26	1.01

TABLE 6. Type 2 processor operating speed.

Parameters	Value			
$V_{dd}(V)$	1.3	1.1	1.0	0.85
$f(MHz)$	400	300	200	100

The simulation environment is built in Matlab version R2016a moreover, the experiments are conducted using hardware platform of Intel(R) Xeon(R), i5-3570 CPU with a clock frequency of 3.5 GHz and 16.0 GB memory, 10 MB cache. Two types of processors, Type 1 (Transmeta Crusoe) and Type 2 (PXA-250) are deployed. Type 1 processor model is acquired from [18] and operates at five different voltage levels in the range from 0.65 V to 0.85 V with 50 mV step. Type 2 is modeled using processor in [51] and it operates at four discrete voltage levels from 0.85 V to 1.3 V with 0.15 V step size. The power and energy consumption for different power modes of these processors are listed in TABLE 4. Where TABLE 5, and TABLE 6 show the operating speed, i.e. different voltage and frequency levels of type 1 and type 2 processors respectively. Furthermore, intlinprog solver for programming ILP problems is also used in the simulation.

B. CITM-VA ENERGY PERFORMANCE

Different results have been generated considering the parameters such as dynamic energy (d), static energy (s), number of tiles or NoC structure and makespan. TABLE 7 shows the dynamic energy consumption of the real-world benchmarks using a different number of tiles. Dynamic plus static energy consumption is summarized in TABLE 8. Where the energy

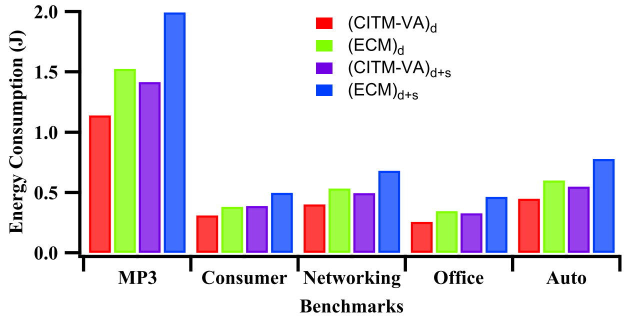

FIGURE 5. 5 × 4 NoC containing 20 tiles.

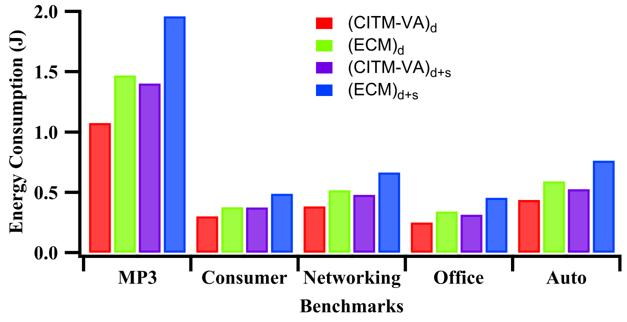

FIGURE 6. 6 × 4 NoC containing 24 tiles.

consumptions at 0.9 × $(Makespan)_{ECM}$ and Makespan Multiplier (MM) ranging from 0.9 to 1.1 with 0.05 step are listed in TABLE 9, and TABLE 10 respectively.

1) 5 × 4 NOC

FIGURE 5 shows the energy performance comparison of 5 real-world benchmarks using 5 × 4 NoC, i.e. 20 tiles. X-axis represents the benchmarks (task graphs) and Y-axis denotes the energy consumption in joules (J). The proposed $CITM - VA$ performs better than ECM in terms of energy savings. Numerically $(CITM - VA)_d$ achieves $\sim$ 23% total energy efficiency over $(ECM)_d$. This energy performance improvement is because unlike $(ECM)_d$ the developed approach $(CITM - VA)_d$ deploys heterogeneous MPSoC architecture while explicitly considers energy performance profiles of the processors and generates a feasible solution with lower energy consumption. Moreover, $(CITM - VA)_d$ performs mapping, scheduling and voltage scaling in an integrated manner to guide the given task mapping problem to a more energy efficient solution. Some task nodes after execution on MPSoC platform may produce idle slack on the processors which is not addressed by $(ECM)_{d+s}$. Contrarily, beside the mapping and voltage scaling $(CITM - VA)_{d+s}$ also deploys DPM strategy to minimize the energy consumption during processor's idle slack period. A $\sim$ 25% increase in the total energy consumption $(CITM - VA)_{d+s}$ occurs compared to $(CITM - VA)_d$ because of the leakage power consumption. Compared to $(ECM)_{d+s}$ the proposed energy management approach $(CITM - VA)_{d+s}$ saves $\sim$ 27% total energy.

TABLE 7. Real-world benchmarks dynamic energy consumption.

Application	20-Tiles		24-Tiles		28-Tiles	
Benchmark	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)
MP3-encoder	1.139	1.524	1.074	1.470	0.971	1.520
Consumer	0.309	0.382	0.302	0.377	0.296	0.371
Networking	0.401	0.533	0.384	0.517	0.358	0.522
Office-automation	0.257	0.346	0.249	0.341	0.244	0.337
Auto-industry	0.448	0.599	0.438	0.590	0.433	0.586

TABLE 8. Real-world benchmarks dynamic+static energy consumption.

Application	20-Tiles		24-Tiles		28-Tiles	
Benchmark	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)
MP3-encoder	1.415	1.994	1.402	1.960	1.361	1.941
Consumer	0.389	0.497	0.374	0.488	0.363	0.483
Networking	0.496	0.679	0.480	0.665	0.465	0.658
Office-automation	0.328	0.463	0.315	0.454	0.306	0.450
Auto-industry	0.549	0.778	0.527	0.762	0.511	0.753

TABLE 9. Real-world benchmarks dynamic+static energy at 0.9 × $(makespan)_{ECM}$.

Application	20-Tiles		24-Tiles		28-Tiles	
Benchmark	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)
MP3-encoder	1.563	—	1.507	—	1.479	—
Consumer	0.411	—	0.401	—	0.409	—
Networking	0.527	—	0.510	—	0.496	—
Office-automation	0.347	—	0.332	—	0.324	—
Auto-industry	0.583	—	0.561	—	0.550	—

TABLE 10. Energy response at different MM values using 28 tiles MPSoC architecture.

Deadline	Benchmarks									
MM	MP3-encoder		Consumer		Networking		Office-automation		Auto-industry	
	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)	$E_{CITM-VA}$ (J)	E_{ECM} (J)
0.90	1.479	—	0.409	—	0.496	—	0.324	—	0.550	—
0.95	1.403	—	0.388	—	0.481	—	0.317	—	0.534	—
1.00	1.361	1.941	0.363	0.483	0.465	0.658	0.306	0.450	0.511	0.753
1.05	1.258	1.873	0.357	0.471	0.450	0.642	0.289	0.435	0.490	0.732
1.10	1.187	1.805	0.345	0.458	0.441	0.624	0.278	0.420	0.473	0.718

2) 6 × 4 NoC

The energy efficiency of $CITM - VA$ is further investigated for all 5 benchmarks when number of tiles are increased to 24 as shown in the FIGURE 6. A reduction in the overall energy consumption occurs when number of tiles are changed from 20 to 24. For example, MP3 consumed 1.139 J, 1.415 J for $(CITM - VA)_d$ and $(CITM - VA)_{d+s}$ respectively using 20 tiles NoC. These energy values are reduced to 1.074 J and 1.402 J for $(CITM - VA)_d$ and $(CITM - VA)_{d+s}$ respectively when 6×4 NoC with 24 tiles is deployed. This energy reduction occurred due to the availability of more number of low energy performance processors for task mapping. Relatively a smaller decrease in energy consumption resulted for *ECM* because it uses MPSoC platform containing homogeneous processors with the same energy performance profiles. The proposed approach $(CITM - VA)_d$ achieves an average $\sim 25\%$ energy improvement compared to $(ECM)_d$. Moreover, $\sim 30\%$ energy efficiency is attained by $(CITM - VA)_{d+s}$ over $(ECM)_{d+s}$.

3) 7 × 4 NOC

FIGURE 7 demonstrates the energy consumption comparison when 7×4 NoC deploying 28 heterogeneous processors on MPSoC architecture is used. A similar pattern like FIGURE 6 is followed by the energy consumption in this case though, the energy savings increased to $\sim$ 25% and $\sim$ 30% for $(CITM - VA)_d$ and $(CITM - VA)_{d+s}$ respectively. For instance $(CITM - VA)_d$ of MP3 reduces to 0.971 J from 1.074 J

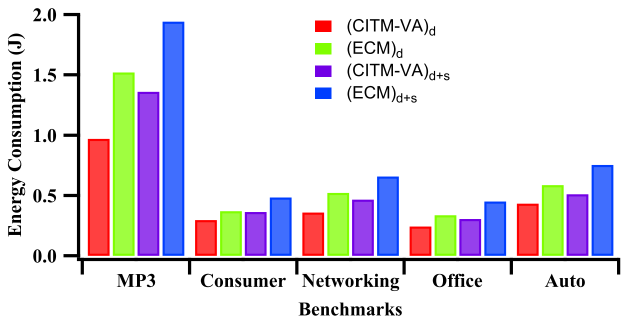

FIGURE 7. 7 × 4 NoC containing 28 tiles.

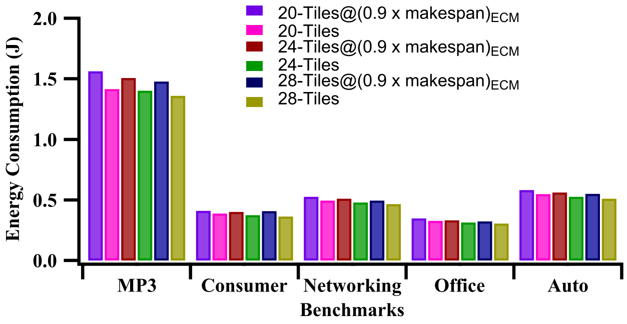

FIGURE 8. Robustness.

and $(CITM - VA)_{d+s}$ minimizes to 1.361 J from 1.402 J when number of tiles are increased from 24 to 28. A similar decrease in the energy consumption is observed for other real-world benchmarks. The energy consumption reduction using $CITM - VA$ is steeper than ECM. Moreover, $(CITM - VA)_d$ and $(CITM - VA)_{d+s}$ improve the energy efficiency $\sim 27\%$ and $\sim 33\%$ respectively.

4) ROBUSTNESS AND QOS

The robustness and QoS of both the schemes are analyzed when the makespan generated by ECM is considered as baseline and multiplied with MM of 0.9 as exhibited in FIGURE 8. The ECM approach can not produce energy consumption approximation at $0.9 \times makespan$ because it does not implement ELFTF strategy to re-arrange the task nodes order according to the deadline set. Therefore, QoS degrades at strict deadline and ECM fails to efficiently perform task mapping and voltage scaling at strict deadlines for real-time applications. Thus, ECM shows no robustness and exibits poor QoS. Contrarily, $CITM - VA@(0.9 \times makespan)_{ECM}$ converges at strict deadlines and produces energy consumption values. Though the energy efficiency reduces to $\sim 18\%$, $\sim 24\%$ for $(CITM - VA)_d$ and $(CITM - VA)_{d+s}$ compared to the normal makespan.

5) ENERGY AND MM

FIGURE 9 demonstrates the total energy consumption of real-world benchmarks for different makespan when 7 × 4 NoC with 28 tiles is used. The total energy consumption of both $CITM - VA$ and ECM decreases when MM value is

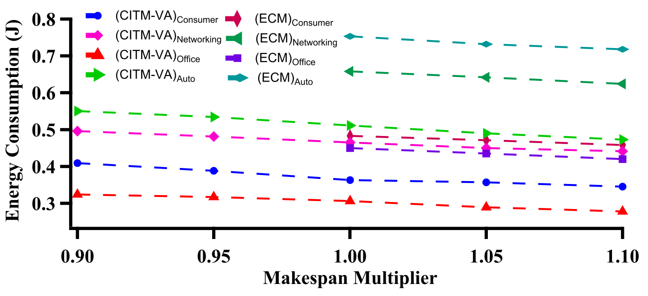

FIGURE 9. Energy and makespan multiplier relation .

increased from 0.9. This reduction of energy consumption is due to the expansion in the common deadline and lower discrete voltage levels can be applied to the tasks and communication nodes. It is worth noticing that $CITM - VA$ generates energy consumption values even at strict deadlines while ECM does not converge and fails to produce output below 1.0 value of MM. Moreover, the total energy consumption of ECM for all benchmarks is higher than $CITM - VA$. So, $CITM - VA$ performs better than ECM at different values of the MM.

VI. CONCLUSION

In this paper, an investigation is performed on contention-aware static mapping and voltage scaling for real-time DAG task set with precedence constraints and individual deadlines using NoC based heterogeneous MPSoC architecture with DVFS-enabled processors. The proposed CITM-VA approach optimizes both the communication and computational energy and performs task mapping, scheduling and voltage scaling in an integrated manner. It adopts ELFTF strategy and generates a prioritized task schedule to adequately utilize the available slack and links. ReMap algorithm is used to efficiently map the task and communication nodes to the resources and discrete voltage levels such that overall energy consumption is reduced. To further improve the energy savings DPM is deployed when an idle processor is in a high-power consumption state. Contention between the communications traversing the same link is eliminated by dedicating the links to higher priority communications. The extensive evaluation results illustrate that compared to state-of-the-art technique ECM, the proposed approach CITM-VA achieves better energy efficiency, i.e. an average $\sim$ 30%. Moreover, it also maintains high QoS and robustness at strict task deadlines with significant energy savings.

REFERENCES

[1] J.-J. Han, M. Lin, D. Zhu, and L. T. Yang, "Contention-aware energy management scheme for NoC-based multicore real-time systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 3, pp. 691–701, Mar. 2015.
[2] S. Mittal and J. S. Vetter, "A survey of CPU-GPU heterogeneous computing techniques," *ACM Comput. Surv.*, vol. 47, no. 4, p. 69, 2015.
[3] H. Okamura and T. Dohi, "Dynamic power management with optimal time-out policies," *IEEE Syst. J.*, vol. 11, no. 2, pp. 962–972, Jun. 2017.
[4] M. A. Kinsy, L. Bu, M. Isakov, and M. Mark, "Designing secure heterogeneous multicore systems from untrusted components," *Cryptography*, vol. 2, no. 3, p. 12, 2018.

[5] T. Chantem, X. S. Hu, and R. P. Dick, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 10, pp. 1884–1897, Oct. 2011.

[6] X. Zhang, K. Huang, M. Yu, X. Jiang, and X. Yan, "BFCO: A BPSO-based fine-grained communication optimization method for MPSoC," *IEEE Access*, vol. 6, pp. 18771–18785, 2018.

[7] U. U. Tariq, H. Wu, and S. A. Ishak, "Energy-aware scheduling of conditional task graphs on NoC-based MPSoCs," in *Proc. 51st Hawaii Int. Conf. Syst. Sci.*, 2018, pp. 5707–5716.

[8] Y.-J. Chen, W.-W. Chang, C.-Y. Liu, C.-E. Wu, B.-Y. Chen, and M.-Y. Tsai, "Processors allocation for MPSoCs with single ISA heterogeneous multi-core architecture," *IEEE Access*, vol. 5, pp. 4028–4036, 2017.

[9] J. Zhou *et al.*, "Thermal-aware correlated two-level scheduling of real-time tasks with reduced processor energy on heterogeneous MPSoCs," *J. Syst. Archit.*, vol. 82, pp. 1–11, Jan. 2018.

[10] L. Torres, P. Benoit, G. Sassatelli, M. Robert, F. Clermidy, and D. Puschini, "An introduction to multi-core system on chip—Trends and challenges," in *Multiprocessor System-on-Chip*. New York, NY, USA: Springer, 2011, pp. 1–21.

[11] A. Prakash, S. Wang, A. E. Irimiea, and T. Mitra, "Energy-efficient execution of data-parallel applications on heterogeneous mobile platforms," in *Proc. 33rd IEEE Int. Conf. Comput. Design (ICCD)*, Oct. 2015, pp. 208–215.

[12] P. K. Valsan, H. Yun, and F. Farshchi, "Taming non-blocking caches to improve isolation in multicore real-time systems," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp. (RTAS)*, Apr. 2016, pp. 1–12.

[13] H.-E. Zahaf, A. El Hassen Benyamina, R. Olejnik, and G. Lipari, "Energy-efficient scheduling for moldable real-time tasks on heterogeneous computing platforms," *J. Syst. Archit.*, vol. 74, pp. 46–60, Mar. 2017.

[14] J. Ceng and R. Leupers, "A methodology for efficient multiprocessor system on chip software development," Ph.D. dissertation, RWTH Aachen Univ., Aachen, Germany, 2011.

[15] E. L. de Souza Carvalho, N. L. V. Calazans, and F. G. Moraes, "Dynamic task mapping for MPSoCs," *IEEE Des. Test Comput.*, vol. 27, no. 5, pp. 26–35, Sep./Oct. 2010.

[16] W. Liu, Z. Gu, J. Xu, X. Wu, and Y. Ye, "Satisfiability modulo graph theory for task mapping and scheduling on multiprocessor systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 8, pp. 1382–1389, Aug. 2011.

[17] S. Sachdeva and P. Panwar, "A review of multiprocessor directed acyclic graph (DAG) scheduling algorithms," *Int. J. Comput. Sci. Commun.*, vol. 6, no. 1, pp. 67–72, 2015.

[18] G. Chen, K. Huang, and A. Knoll, "Energy optimization for real-time multiprocessor system-on-chip with optimal DVFS and DPM combination," *ACM Trans. Embedded Comput. Syst.*, vol. 13, no. 3s, p. 111, 2014.

[19] W. Zhang, E. Bai, H. He, and A. M. K. Cheng, "Solving energy-aware real-time tasks scheduling problem with shuffled frog leaping algorithm on heterogeneous platforms," *Sensors*, vol. 15, no. 6, pp. 13778–13804, 2015.

[20] Y. Wang, D. Liu, M. Wang, Z. Qin, and Z. Shao, "Optimal task scheduling by removing inter-core communication overhead for streaming applications on MPSoC," in *Proc. 16th IEEE Real-Time Embedded Technol. Appl. Symp. (RTAS)*, Apr. 2010, pp. 195–204.

[21] J. Huang, C. Buckl, A. Raabe, and A. Knoll, "Energy-aware task allocation for network-on-chip based heterogeneous multiprocessor systems," in *Proc. 19th Int. Euromicro Conf. Parallel, Distrib. Netw. Based Process. (PDP)*, Feb. 2011, pp. 447–454.

[22] U. U. Tariq and H. Wu, "Energy-aware scheduling of conditional task graphs with deadlines on MPSoCs," in *Proc. IEEE 34th Int. Conf. Comput. Design (ICCD)*, Oct. 2016, pp. 265–272.

[23] Y. Wang, H. Liu, D. Liu, Z. Qin, Z. Shao, and E. H.-M. Sha, "Overhead-aware energy optimization for real-time streaming applications on multiprocessor system-on-chip," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 16, no. 2, p. 14, 2011.

[24] D. Li and J. Wu, "Energy-efficient contention-aware application mapping and scheduling on NoC-based MPSoCs," *J. Parallel Distrib. Comput.*, vol. 96, pp. 1–11, Oct. 2016.

[25] F. Ferrandi, C. Pilato, D. Sciuto, and A. Tumeo, "Mapping and scheduling of parallel c applications with ant colony optimization onto heterogeneous reconfigurable mpsocs," in *Proc. Asia South Pacific Design Autom. Conf.*, Jan. 2010, pp. 799–804.

[26] S. A. Murtza, A. Ahmad, M. Y. Qadri, N. N. Qadri, and J. Ahmed, "Optimizing energy and throughput for MPSoCs: An integer particle swarm optimization approach," *Computing*, vol. 100, no. 3, pp. 227–244, 2018.

[27] K. Srinivasan and K. S. Chatha, "Integer linear programming and heuristic techniques for system-level low power scheduling on multiprocessor architectures under throughput constraints," *Integration*, vol. 40, no. 3, pp. 326–354, 2007.

[28] S.-H. Kang, H. Yang, S. Kim, I. Bacivarov, S. Ha, and L. Thiele, "Static mapping of mixed-critical applications for fault-tolerant mpsocs," in *Proc. 51st ACM/EDAC/IEEE Design Automat. Conf. (DAC)*, Jun. 2014, pp. 1–6.

[29] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *J. Scheduling*, vol. 12, no. 4, p. 417, 2009.

[30] T. K. H. Nguyen, "Low power architecture for fall detection system," Ph.D. dissertation, Université Nice Sophia Antipolis, Nice, France, 2015.

[31] M. Ahmadi, W. J. Gross, and S. Kadoury, "A real-time remote video streaming platform for ultrasound imaging," in *Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Aug. 2016, pp. 4383–4386.

[32] I. Ahmed, A. Ahmad, F. Piccialli, A. K. Sangaiah, and G. Jeon, "A robust features-based person tracker for overhead views in industrial environment," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1598–1605, Jun. 2018.

[33] M. B. Ayed and M. Abid, "An automated surveillance system based on multi-processor system-on-chip and hardware accelerator," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 9, pp. 59–66, 2017.

[34] K. Muhammad, J. Ahmad, I. Mehmood, S. Rho, and S. W. Baik, "Convolutional neural networks based fire detection in surveillance videos," *IEEE Access*, vol. 6, pp. 18174–18183, 2018.

[35] A. Safaei, Q. M. J. Wu, and Y. Yang, "System-on-a-chip (SoC)-based hardware acceleration for foreground and background identification," *J. Franklin Inst.*, vol. 355, no. 4, pp. 1888–1912, 2018.

[36] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, "Internet of Things security and forensics: Challenges and opportunities," *Future Gener. Comput. Syst.*, vol. 78, pp. 544–546, Jan. 2018.

[37] Y. Liu, K. A. Hassan, M. Karlsson, O. Weister, and S. Gong, "Active plant wall for green indoor climate based on cloud and Internet of Things," *IEEE Access*, vol. 6, pp. 33631–33644, 2018.

[38] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of Things for smart healthcare: Technologies, challenges, and opportunities," *IEEE Access*, vol. 5, pp. 26521–26544, 2017.

[39] W. Wang, S. De, R. Toenjes, E. Reetz, and K. Moessner, "A comprehensive ontology for knowledge representation in the Internet of Things," in *Proc. IEEE 11th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Jun. 2012, pp. 1793–1798.

[40] G. White, V. Nallur, and S. Clarke, "Quality of service approaches in IoT: A systematic mapping," *J. Syst. Softw.*, vol. 132, pp. 186–203, Oct. 2017.

[41] J. Li, Y. Bai, N. Zaman, and V. C. M. Leung, "A decentralized trustworthy context and QoS-aware service discovery framework for the Internet of Things," *IEEE Access*, vol. 5, pp. 19154–19166, 2017.

[42] Y. Cheng, L. Zhang, Y. Han, and X. Li, "Thermal-constrained task allocation for interconnect energy reduction in 3-D homogeneous MPSoCs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 2, pp. 239–249, Feb. 2013.

[43] S. Tosun, "Energy- and reliability-aware task scheduling onto heterogeneous MPSoC architectures," *J. Supercomput.*, vol. 62, no. 1, pp. 265–289, 2012.

[44] D. Shin and J. Kim, "Communication power optimization for network-on-chip architectures," *J. Low Power Electron.*, vol. 2, no. 2, pp. 165–176, 2006.

[45] Y. Wang, Z. Shao, H. C. B. Chan, D. Liu, and Y. Guan, "Memory-aware task scheduling with communication overhead minimization for streaming applications on bus-based multiprocessor system-on-chips," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1797–1807, Jul. 2014.

[46] C.-L. Chou and R. Marculescu, "Contention-aware application mapping for network-on-chip communication architectures," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, Oct. 2008, pp. 164–169.

[47] J. Singh, S. Betha, B. Mangipudi, and N. Auluck, "Contention aware energy efficient scheduling on heterogeneous multiprocessors," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1251–1264, May 2015.

[48] A. Andrei, P. Eles, Z. Peng, M. T. Schmitz, and B. M. A. Hashimi, "Energy optimization of multiprocessor systems on chip by voltage selection," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 3, pp. 262–275, Mar. 2007.

[49] M. F. Reza, D. Zhao, and H. Wu, "Task-resource co-allocation for hotspot minimization in heterogeneous many-core nocs," in *Proc. 26th Ed. Great Lakes Symp. VLSI*, 2016, pp. 137–140.

[50] A. Asad, A. Dorostkar, and F. Mohammadi, "A novel power model for future heterogeneous 3d chip-multiprocessors in the dark silicon age," *EURASIP J. Embedded Syst.*, vol. 1, no. 1, p. 3, 2018.

[51] R. Jejurikar, C. Pereira, and R. Gupta, "Leakage aware dynamic voltage scaling for real-time embedded systems," in *Proc. 41st Annu. Design Autom. Conf.*, 2004, pp. 275–280.

HAIDER ALI received the master's degree in electronic systems design engineering from Manchester Metropolitan University, U.K. He is currently pursuing the Ph.D. degree with the Department of Electronics, Computing and Mathematics, University of Derby. He was a Lecturer at COMSATS IIT from 2011 to 2016. He has published more than 10 research articles in prominent journals and conferences. He is currently working on energy efficient algorithms for task mappings on modern embedded systems for real-time applications. His research area of interest is electronic and biomedical systems design, image processing, and embedded systems.

UMAIR ULLAH TARIQ received the master's degree from the University of Engineering and Technology, Taxila, Pakistan, and the Ph.D. degree in computer science and engineering from the University of New South Wales, Sydney, NSW, Australia. He is currently a Lecturer with the Department of Electrical Engineering, COMSATS Institute of Information Technology, Abbottabad, Pakistan. He has published several research papers in prominent IEEE conferences on task scheduling using MPSoCs. His current research interests include energy-aware scheduling, digital image processing, and computer vision.

YONGJUN ZHENG received the B.Sc., M.Sc., and Ph.D. degrees in computer science from Nottingham Trent University. He was a Researcher at the University of Cambridge and Middlesex University before he joined York St John University. He is currently a Lecturer in computer science with Derby University and also with York St John University. His primary research interest concerns HCI, including data visualization, mobile computing, and intelligent systems. He has been the co-investigator of few funding projects and was a reviewer of a few international journals and the program committee member of several international conferences.

XIAOJUN ZHAI received the B.Sc. degree from the North China University of Technology, China, in 2006, and the M.Sc. degree in embedded intelligent systems and the Ph.D. degree from the University of Hertfordshire, U.K., in 2009 and 2013, respectively. He is currently a Lecturer with the School of Computer Science and Electronic Engineering, University of Essex. His research interests mainly include the design and implementation of the digital image and signal processing algorithms, custom computing using FPGAs, embedded systems, and hardware/software co-design. He is a BCS Member and HEA Fellow.

LU LIU received the Ph.D. degree from the Surrey Space Centre, University of Surrey (funded by DIF DTC), and the M.Sc. degree in data communication systems from the Department of Electronic and Computer Engineering, Brunel University. He is currently the Head of the Department of Electronics, Computing and Mathematics and a Professor of Distributed Computing. He has authored over 170 scientific publications in reputable journals. His research interests are in areas of data analytics, service computing, cloud computing, information systems, and Internet-of-Things. He serves as an Editorial Board Member for six international journals and the Guest Editor for 12 international journals.