Energy-aware Scheduling of Streaming Applications on Edge-devices in IoT based Healthcare

Umair Ullah Tariq, Haider Ali, Lu Liu, James Hardy, Muhammad Kazim, and Waqar Ahmed

***Abstract*—The reliance on Network-on-Chip (NoC) based Multiprocessor Systems-on-Chips (MPSoCs) is proliferating in modern embedded systems to satisfy the higher performance requirement of multimedia streaming applications. Task level coarse grained software pipeling also called re-timing when combined with Dynamic Voltage and Frequency Scaling (DVFS) has shown to be an effective approach in significantly reducing energy consumption of the multiprocessor systems at the expense of additional delay. In this paper we develop a novel energy-aware scheduler considering tasks with conditional constraints on Voltage Frequency Island (VFI) based heterogeneous NoC-MPSoCs deploying re-timing integrated with DVFS for real-time streaming applications. We propose a novel task level re-timing approach called R-CTG and integrate it with non linear programming based scheduling and voltage scaling approach referred to as ALI-EBAD. The R-CTG approach aims to minimize the latency caused by re-timing without compromising on energy-efficiency. Compared to R-DAG, the state-of-the-art approach designed for traditional Directed Acyclic Graph (DAG) based task graphs, R-CTG significantly reduces the re-timing latency because it only re-times tasks that free up the wasted slack. To validate our claims we performed experiments on using 12 real benchmarks, the results demonstrate that ALI-EBAD out performs CA-TMES-Search and CA-TMES-Quick task schedulers in terms of energy-efficiency.**

***Index Terms*—IoT, Video-streaming, CTGs, Re-timing, MPSoCs, Edge-device, Scheduling, Energy-efficiency, Prologue.**

I. Introduction

HEALTHCARE is one of the fastest growing industries with an enormous potential for enhancement from the employment of technologies such as the Internet-of-Things (IoT), cloud computing, and mobile devices. An IoT based ubiquitous healthcare system may provide patients the opportunity to lead a more independent life without the constant need for qualified medical staff to monitor their health. The advanced healthcare system using the IoT not only provides an accurate medical data but also integrates an alarm system for emergency situations. The IoT is making healthcare more accessible and responsive to the needs of most users anywhere and at anytime. The IoT consists of a combination of sensors and actuators including, analogue devices, and cameras. Utilizing features such as video-streams transmitted via the Internet, these systems are able to monitor the healthcare needs for anyone. This is considered most beneficial to people with increases support needs such as elderly, inform or those with alternative abilities. The video and data streams are stored securely in the cloud but available for access by the right people when needed. An abstract IoT based healthcare system architecture is demonstrated in Fig. (1) [1], [2].

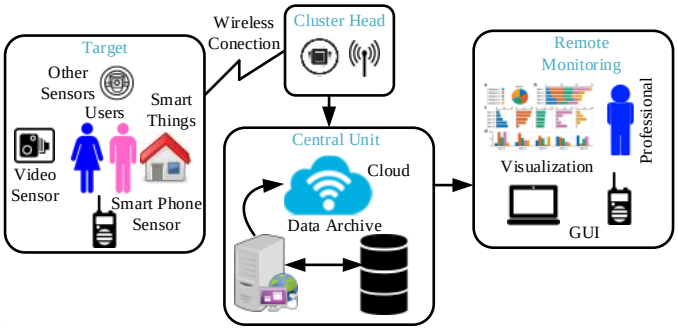


Fig. (1) IoT based Healthcare System Architecture

Thanks to the advancements in technology, multimedia data-streaming and live video-streaming have provided significant positive impacts on healthcare by enabling professional, real-time virtual healthcare assistance in places which was not possible before the era of the Internet. Multimedia applications are providing a new baseline for advanced healthcare. Literature predicted there to be approximately 50 billion interconnected IoT digital devices by 2020 [3]. Multimedia content will be approximately 80% of the total Internet data traffic by 2021 and healthcare will be a significant proportion of this data. The multimedia data content is streamed over the network in an encoded form with the video displayed to the end user and/or professional either in a recorded or pre-recorded manner. Real-time video-streaming applications in IoT are periodic in nature as they tend to be executed repeatedly. In IoT, video streams are usually compressed to reduce the video size and achieve better network load balancing. The MPEG-encoder is executed numerous times for the whole video stream. The multimedia streaming data can be represented by Conditional Task Graph (CTG), the tasks within the CTG are dependent on each other [4]. Prime examples of real-time IoT based multimedia streaming applications in healthcare include human gait analysis, telemonitoring and fall detection in people with infirmities [5], [6].

In real-time streaming applications, tasks are dependent on each other thus, the slack within the processors of the MPSoC architecture is not efficiently utilized. Re-timing is a powerful

Umair Ullah Tariq is with the School of Engineering and Technology, Central Queensland University, Sydney, Australia. e-mail: u.tariq@cqu.edu.au

Haider Ali and James Hardy are with the Department of Electronics, Computing and Mathematics, University of Derby, Derby, UK. e-mail: h.ali@derby.ac.uk

Lu Liu is with the School of Informatics, Leicester University, Leicester, UK. e-mail: l.liu@leicester.ac.uk

Muhammad Kazim is with the Faculty of Computing, Engineeringand Media, De Montfort University, Leicester, UK, e-mail: Muhammad.kazim@dmu.ac.uk

Waqar Ahmed is with the Department of Computer Engineering, UET, Taxila, Pakistan. e-mail: waqar.ahmad@uettaxila.edu.pk

technique applied at the application level to transform the intra-period dependencies between tasks by regrouping the tasks from different periods. Concisely, re-timing transforms a dependent task model into an independent task model to efficiently utilize the resources [7]. However, the process of re-timing adds an unwanted delay called prologue during video-streaming. Ideally there is a need of an efficient video-streaming where the video starts playing with a reduced prologue.

Creation and manipulation of multimedia data is computationally expensive due to intensive processes such as video encoding, compression, and Fourier Transform (FT). Consequently, Multiprocessor System-on-Chips (MPSoCs) have become an essential element of the modern embedded systems for real-time multimedia data processing due to their higher performance, reliability and exceptional Quality-of-Service (QoS) [8], [9]. Xilinx Zynq® UltraScale+™ MPSoCs and Tilera TILE-Gx72™ are few of the well-known high performance computing architectures used in digital systems for healthcare Examples of the medical application of multiprocessor systems include a real-time video-streaming system [10] developed to remotely monitor the human ultrasound examinations. The ultrasound streams are transmitted wirelessly to a remote location where the information is accessed and analyzed by a medical specialist. Similar works in [11], [12] used heterogeneous MPSoCs and enhanced the image quality for the ultrasound imaging. These developments improved the overall performance and reduced latency. In a further example, a human fall detection mechanism is presented [13] using a ZYNQ MPSoC platform to perform various operations such as segmentation, feature extraction, filtering and recognition for differentiating different types of fall.

Data extensive real-time applications are increasing in IoT, increasing numbers of processors elements are therefore desirable in MPSoCs design to meet the performance needs [9], [14], [15]. According to the International Technology Roadmap for Semiconductors (ITRS), MPSoCs will consist of hundreds of processors in the near future. In this case, traditional bus-based MPSoCs would become a computational bottleneck due to their higher inter-element communication requirement leading to higher congestion and poor scalability. Alternatively, Network-on-Chip (NoC) based communication available in some MPSoCs can offer improved scalability with higher flexibility [16], [17]. Recently Voltage Frequency Island (VFI), Globally Asynchronous Locally Synchronous (GALS) introduced to NoC interconnect, where the tiles/processors are partitioned into islands. Each island in an MPSoC operates at its own frequency and supply voltage to minimize the total energy consumption [18]. These attributes lead to higher throughput and lower hardware complexity and make VFI based heterogeneous NoC-MPSoC (VFI-NoC-HMPSoC) the most suitable choice of computing platform for data extensive applications [19].

Energy consumption reduction in digital systems using MPSoCs for the IoT based healthcare is an important research aspect because higher energy consumption produces an increased carbon footprint [9], [18]. Proper task scheduling approaches can reduce energy consumption and increase the performance and reliability of an embedded system [20]. Task scheduling is NP-hard problem therefore, different heuristics have been developed to achieve energy-efficient solutions [21]. Real-time multimedia streaming applications, as used in healthcare, are typical illustrations of static task scheduling. Dynamic Voltage and Frequency Scaling (DVFS) is a conventional approach integrated with scheduling to minimize the energy consumption of MPSoC computing architectures [22]. DVFS efficiently utilizes the available slack within the processors by dynamically reducing the supplied voltage/clock frequency without violating the tasks deadline constraint subsequently, it minimizes the overall power consumption [7].

In this paper, we investigate an energy-efficient static scheduling deploying VFI-NoC-HMPSoC for a set of periodic tasks with conditional precedence constraints representing a real-time periodic streaming application. Our contributions and innovations include as follows:

1) We develop a novel energy-aware static scheduler considering tasks with conditional constraints using VFI-NoC-HMPSoC computing architecture deploying a re-timing technique integrated with DVFS for the IoT based real-time streaming applications in healthcare.
2) We present a non linear programming (NLP) based scheduling and voltage scaling approach which we refer to as $ALI-EBAD$. This performs task scheduling and voltage scaling in an integrated manner to steer the task scheduling towards a more energy-efficient solution.
3) We propose a novel task level coarse-grained software pipelining approach called Re-timed CTG (R-CTG). This significantly reduces the re-timing latency compared to R-DAG [23], without an increase in energy consumption.
4) We show by experiments that that ALI-EBAD deploying VFI-NoC-HMPSoC achieves an average energy-efficiency of $\sim 20\%$ over CA-TMES-Search [24] and $\sim 25\%$ over CA-TMES-Quick [24]. The energy-efficiency increases significantly when R-CTG is integrated with ALI-EBAD and attains an average energy saving of $\sim 40\%$ and $\sim 45\%$ respectively. The R-CTG efficiently reduces the prologue/latency by 50% when compared with state-of-the-art re-timing technique R-DAG [23].

The remainder of the paper is organized as follows: Section II discusses the related work performed so far on task scheduling using multiprocessors. Section III presents application, system, and energy models used in the simulations. Section IV explains our novel offline pipelined scheduling. Section V presents experimental results followed by the conclusion of this paper in Section VI.

II. Literature Review

Multiprocessor systems are becoming de-facto computing platforms due to their excellent high-performance and exceptional QoS. For this reason, different research studies have deployed multiprocessor computing architectures for energy-aware task scheduling.

Olafsson introduced one of the first dynamical models for the scheduling of tasks on heterogeneous multiprocessor systems [25]. Aydin et al. applied DVFS to determine the optimal

voltage levels for the tasks and developed an algorithm called Earliest Deadline First (EDF) to a generate feasible tasks schedule [26]. Tosun [27] used Integer Linear Programming (ILP) to decrease the computational energy consumption of heterogeneous MPSoC architectures by assigning accurate voltage levels to the tasks. The authors also developed an energy-aware heuristic integrated with EDF technique for efficient task ordering. Kumar and Vidyarthi [28] combined task mapping and discrete voltage levels assignment within the single optimization loop of Genetic Algorithm (GA) and compared the results in terms of energy-savings with Genetic Algorithm-Struggle (GA-ST). Recently Dziurzanski and Singh designed a feedback control- based task scheduling called Admission Control Algorithm (ACA) and performed schedulability analysis to determine the tasks which are expected to violate deadline constraints [29]. Though the aforementioned task scheduling heuristics presented in [26], [27], [28], [29] efficiently reduced energy consumption of the multiprocessor computing systems, these investigations only considered independent task graphs i.e. tasks without precedence constraints.

Other researchers investigated scheduling problems integrated with DVFS for tasks with precedence constraints to reduce the power overhead. For instance, Wang et al. formulated the scheduling problem as ILP and reduced the computational and inter-processor communication overheads of heterogeneous MPSoC for streaming applications. The authors obtained a solution with a possible minimum schedule length using ILP based algorithm and minimized the wasted slack within the schedule deploying DVFS [30]. Chen et al. [31] applied Mixed Integer Linear Programming (MILP) on NoC-MPSoC and developed a heuristic for generating a non-preemptive task schedule while applying discrete voltage to each task. Ali et al. developed a meta-heuristic called Contention-aware Integrated Task Mapping and Voltage Assignment (CITM-VA) for static task mapping and performed task ordering using Earliest Latest Finish Time First (ELFTF) [9]. While the investigations performed for task scheduling problems on MPSoC in [30], [31], [9] only focus on dependent tasks represented by Directed Acyclic Graph (DAG).

The research studies in [32], [33], [34], [16], [35] explored energy-efficient scheduling for CTGs. For example, Shin and Kim [32] developed a Non Linear Programming (NLP) based heuristic for assigning optimal discrete voltage levels to each task in order to reduce the computational energy consumption. Wu et al. [33] presented an algorithm which deploys the schedule table generated by Eles et al. [34] for calculating the available slack in the processors and assigns voltage levels to each task using a voltage scaling algorithm. Tariq and Wu [16] scheduled the tasks represented by CTGs on homogeneous MPSoC and formulated the scheduling problem as NLP. An algorithm called Iterative Offline Energy-aware Task and Communication Scheduling (IOETCS) is used to perform scheduling and voltage scaling in an integrated manner. IOETCS used the Earliest Successor-Tree-Consistent Deadline First (ESTCDF) algorithm for generating an initial task schedule and then applies voltage scaling using ILP [35]. Each of these research papers only investigated energy-aware conditional task scheduling on single processor per VFI.

Recently, task scheduling deploying VFI based MPSoCs have been explored in other studies that use bus as a communication interconnect. For example, Pagani et al. [36] presented a Single Frequency Approximation (SFA) algorithm for optimal voltage assignment to the processor islands in MPSoC architecture. The SFA is integrated with Dynamic Programming Mapping Algorithm (DPMA) to increase the energy-efficiency and to minimize the running time. Liu and Guo [37] developed an algorithm called Voltage Island Largest Capacity First (VILCF) for task scheduling. The VILCF reduces the energy consumption by fully utilizing an island that is already active before activating other islands. Han et al. [24] mapped tasks on the processors of the islands and communications on the NoC to reduce the overall makespan and inter-VFI communication. The authors developed two contention and energy-aware task mapping and edge scheduling heuristics called CA-TMES-Quick and CA-TMES-Search for assigning tasks to processors and edges on NoC. Tariq et al. [38] developed a meta-heuristic for energy-efficient and contention-aware dependent tasks with precedence constraints on VFI-NoC-HMPSoC. Gammoudi et al. [39] scheduled periodic tasks on homogeneous NoC-VFI-MPSoC architectures deploying well known EDF task ordering policy. Though these investigations reduced the energy consumption by utilizing appropriate task mapping and scheduling, they did not consider re-timing at the task level to further minimize the total energy consumption.

Researchers in other investigations have developed algorithms to transform the intra-period dependencies in DAG tasks into inter-period dependencies using re-timing integrated with DVFS to achieve higher energy-efficiency. Wang et al. [23] transformed the dependent tasks into independent task models deploying an algorithm called R-DAG, and used a heuristic termed as GeneS for voltage assignment plus task mapping. Wang et al. [40] reduced the inter-processor communication overhead using re-timing by deploying an algorithm called Joint Computation and Communication Task Scheduling (JCCTS). The JCCTS combined with DVFS reduces the computational energy consumption for real-time tasks. In another research study, Wang et al. [41] reduced both memory and communication overhead using algorithms called Memory-Aware Optimal Task Scheduling (MAOTS) and Heuristic Memory-Aware Task Scheduling (HMATS). These scheduling approaches fail to implement re-timing for tasks represented by CTGs on VFI-NoC-MPSoC architectures.

Concisely, to the best of our knowledge, no prior work has been done that focuses on energy-aware scheduling of tasks with conditional constraints represented by CTGs on VFI-NoC-HMPSoC deploying re-timing combined with DVFS technique.

III. Models and Definitions

In this section, CTG is explained followed by the discussion of our computing architecture deployed for energy-aware task scheduling, and finally the energy model is presented that is used to carry out the simulations.

A. Application Model

The application in our model is represented by a CTG. A CTG is a weighted DAG, $G(V, E, A, W, X)$ [16]. $V = \{v_1, v_2, \ldots, v_n\}$ shows a set of tasks while each task has a certain execution time represented by the number of clock cycles $NC_{i,k}$ on a processor pe_k, a common period T and an individual soft deadline $d_i \leq T$. $E \subseteq V \times V$ is a set of directed edges each denoting the dependency between the two tasks. A is a set of triplets $(e_i, c_i, p(c_i))$, where $e_i \in E$, and c_i and $p(c_i)$ represent the condition associated with e_i and its probability [42], respectively. X is a set of edge weights. An edge weight $\chi_s \in X$ of an edge $e_s = (v_i, v_j)$ denoted the communication volume in bits from task v_i to task v_j.

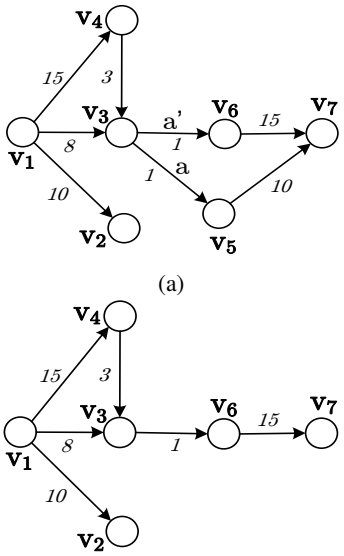


Fig. (2) (a) CTGs (b) A scenario of CTG, G

A *scenario* of a CTG G is a sub-graph of G formed by all the tasks in a complete execution trace of the task set. Fig. (2b) shows a scenario of the CTG G in Fig. (2a) with $a = true$. Given a CTG G_i, an *activation space* AS is a set of all the possible conditions each of which corresponds to a unique scenario. For example, the activation space of the CTG G shown in Fig. (2a) is $AS = \{a, a'\}$. The probability of a scenario $s \in AS$, represented by $p(s)$, is calculated as $p(s) = \prod_{c \in s} p(c)$ where c is a condition that belongs to the scenario s, and $p(c)$ is the probability when c is true. Associated with each task is its *activation probability*. The activation probability of a task is the probability with which the task can be executed. Let S_j be the set of scenarios to which a task v_j belongs to. The activation probability of v_j is calculated as follows:

$$p(v_j) = \sum_{s \in S_j} p(s) \tag{1}$$

B. System Model

We consider a NoC based VFI-MPSoC with M processors $P = \{p_1, p_2, p_3, \ldots p_M\}$ as demonstrated in Fig. (3). In a single tile there is a processor, local memory and network interface. In each tile, the processor executes tasks, memory

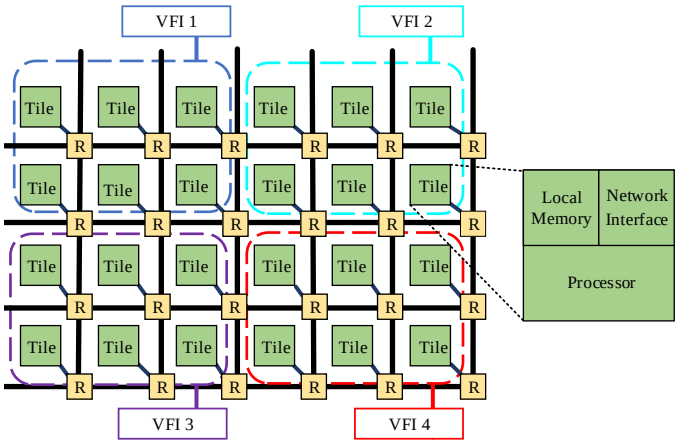


Fig. (3) VFI-NoC-MPSoC Architecture

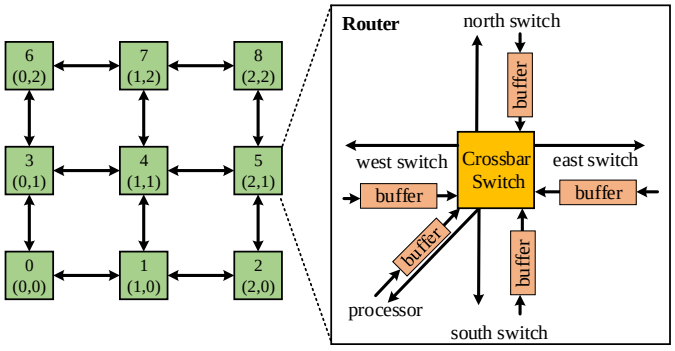


Fig. (4) 2D-Mesh Topology NoC

holds the scheduled tasks and the network interface connects the processor with the router (R) of the mesh network. The processors of the target computing architecture are grouped into a set $C = \{c_1, c_2, c_3, \ldots c_m\}$ of m heterogeneous VFIs. Heterogeneous VFIs have processors with different energy performance profiles. That is one VFI contains lower performance but higher energy-efficient processors while other VFI consists of higher performance but lower energy-efficient processors. Each VFI, $c_i \in C$ of the computing system contains k number of homogeneous processors (processors with same energy performance profile). A single VFI can operate independently at n discrete voltage and frequency levels, $\{(V_{dd_1}, f_1), (V_{dd_2}, f_2), (V_{dd_3}, f_3), \ldots, (V_{dd_n}, f_n)\}$ while a common supply voltage is shared by intra-VFI processors and routers (R).

1) Topology: We consider a 2D-mesh topology NoC for the communication architecture of the MPSoC as exhibited in Fig. (4). Each tile of the NoC based VFI-MPSoC is associated with a router. The NoC mesh contains N_R rows and N_C columns. Thus, the total number of processors in the NoC based VFI-MPSoC is equal to $N_R \times N_C$. Each router possesses five ports, out of which four ports are deployed to communicate with the neighbour routers while one port is used for communicating with the processor. A link connects two routers and/or a router with a processor. All links are identical, full duplex and have the same band width, b_w.

2) Switching Technique: We assume Virtual cut-through (VCT) switching. VCT is one of the most popular packet

switching techniques for NoC communications. In VCT routing the buffer size is large and the entire packet is sent to the next node thus, VCT has lower latency and higher link utilization and lesser packet blocking probability

3) Routing Technique: We adopt XY routing which is the most popular deterministic routing technique on NoC. Routing in a network decides the path of a packet from source to the destination router/node. The XY routing specifically targets 2D-mesh topology and it is the most suitable option for mesh topology networks. Moreover, XY routing is a simple routing but an effective approach additionally it is not prone to deadlock occurrence. In XY routing the packets at the routers are first routed in x-direction and then in the y-direction.

C. Offline Schedule

We consider periodic applications. Periodic applications have a property that they execute repeatedly. Hence, the offline schedule of a CTG is a repeated pattern for the execution of one period of the corresponding periodic conditional dependent tasks. In this work offline schedule consists of both task to processor allocation step and control step assignment. In task to processor allocation step we decide which processor should execute the task and in the control step assignment we decide when to start the task/communication. The key notations we have used in this paper are listed in TABLE (I).

Suppose T is the period of the application then, T indicates the deadline of the schedule, and the schedule must complete within T. We use $\rho(v_i)$ and $\Delta t(v_i)$ to represent respectively the start and the execution time of a task node v_i. Similarly $\rho(v_j, L_k)$ and $\Delta t(v_i, L_k)$ represent respectively the start and transmission time of a communication node v_j on link L_k. $\zeta(v_i)$ represents the finish time of a node v_i.

Given the start time $\rho(v_i)$ of a node in the first period its start time $\rho(v_i)^l$ in the l^{th} period is $\rho(v_i)^l = \rho(v_i) + (l-1)T$, $l \geq 0$. In section IV-A we discuss in detail our offline scheduling and voltage scaling approach. The energy model that is used in our simulations is also discussed in section IV-A.

TABLE (I) Important Notations and Explanation

Notation	Explanation
CTG	Conditional Task Graph
G	An instance of CTG
RT	Re-timing function
v_i	Task node
$RT(v_i)$	Number of periods of v_i reschedule in prologue
(v_i, v_j)	Edge
pe	Processor
RT_{max}	Maximum re-timing value
$prologLatency$	Prologue latency
$\zeta(v_i)$	Finish time of v_i
T	Period of the application
$\rho(v_i)$	Start time of a task node v_i
$\Delta t(v_i)$	Execution time of a task node v_i
$\rho(v_j, L_k)$	Start time of a communication node v_j on link L_k
$\Delta t(v_i, L_k)$	Transmission time of a communication node v_j on link L_k
L_k	k^{th} Link
G_s	Schedule graph
V_s	Set of all scheduled task nodes
V_s^*	Set of communication nodes
E_s	Set of edges
$K1, K2, K6, V_{th_1}$	Circuit dependent constants
L_d	Logic Depth
α	Velocity saturation, $(1.4 \leq \alpha \leq 2)$
NCC	Number of clock cycles
f_u	Router sender frequency
f_v	Router receiver frequency
$IPred(v_i)$	Set of immediate predecessors of v_i
$\Pi(v_i, pe_k)$	Set of task nodes concurrent to v_i
$\Pi(v_i, L_\gamma)$	Set of communication nodes concurrent to v_i
V_{dd}^{min}	Minimum supply voltage
V_{dd}^{max}	Maximum supply voltage
E_i	Energy consumed in execution of a task v_i
C_{eff_k}	Effective switched capacitance
L_g	Number of logic gates
v_{bs}	Body-bias voltage
I_{jn}	Leakage current
E_u	Energy consumed in execution of communication node v_u

IV. Schedule-Aware Pipelining

In this section we discuss our coarse-grained task level pipelining (re-timing) approach. Before we start explaining our proposed re-timing approach we briefly explain re-timing.

The notion of re-timing was originally introduced by [43] to reduce the synchronous circuit cycle period. Recently, [23], [44], [45] extended re-timing to schedule applications represented by a classical DAG (Directed Acyclic Graph) task model on MPSoCs and is defined as follows:

Definition 1: Given a CTG G, re-timing is a function $RT : V \mapsto \mathbb{Z}$, that maps each node $v_i \in G$ to an integer $RT(v_i)$, where $RT(v_i)$ is the number of periods of v_i reschedule in the prologue. Re-timing v_i once if it is legal, reschedules one period of v_i into the prologue.

From the point of view of the program, re-timing regroups loop body such that some or all dependencies within a period are transformed into inter-period dependencies. The re-timing function is valid if no reference is made to the data from the future period. The definition of a valid re-timing function is:

Definition 2: Given a CTG G, for each edge $(v_i, v_j) \in G$ and $v_i, v_j \in G$, the re-timing function RT is said to be valid if $RT(v_i) - RT(v_j) \geq 0$.

If $RT(v_i) - RT(v_j) < 0$, the re-timing function is illegal because this condition implies a reference to unavailable data from the future.

Fig. (5)(a) shows the schedule of the first three periods of the CTG shown in Fig. (6)(a). TABLE II shows the execution time and energy consumption of the tasks of the CTG in Fig. (6)(a). The application is scheduled on MPSoC that consists of two processors pe_1 and pe_2. TABLE II shows the energy consumption and execution time of each task on the two processors. Compared to pe_1, pe_2 is more energy efficient.

Fig. (5)(a) shows a schedule generated by CA-TMES-Search. Notice that CA-TMES-Search fails to efficiently utilize the more energy-efficient processor pe_2 because it favours the processor on which the task can start the earliest. pe_2 remains idle until v_1 completes execution on pe_1. This is because of the intra-period dependency between v_1 and v_2. CA-TMES-Search cannot utilize this slack.

Fig. (5)(c) shows the schedule generated by our approach. Notice that if intra-period dependencies can be transformed

into inter-period dependencies then the wasted slack can be utilized. This can be obtained by regrouping tasks from different periods with computation and communication node rescheduling. As each task is periodic, in Fig. (5)(c), we reschedule periodic task v_1 and execute it one period before v_2 and v_3. The newly added period is called the prologue. In this way the data required by v_2 and v_3 is available at the start of each period, therefore v_2 can start early on pe_2. Consequently, task v_4 can be scheduled on pe_2. Since our approach can utilize the available resources more efficiently hence, it is able to generate more energy-efficient schedule. The energy consumption of the schedule in Fig. (5)(a) is $7.5nJ$ where as the schedule in Fig. (5)(c) consumes $7nJ$. Our approach can further reduce the energy to $6nJ$ if the MPSoC has another energy-efficient processor like pe_2.

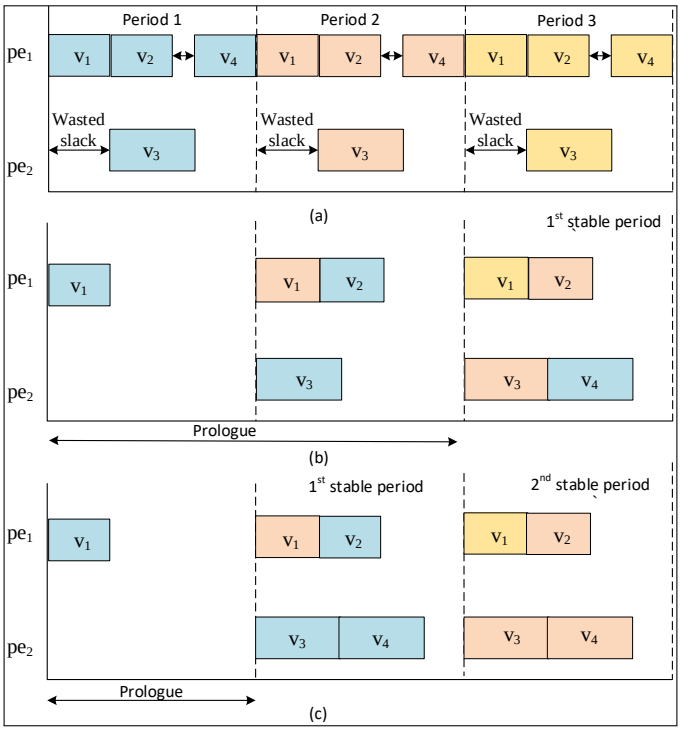


Fig. (5) Periodic Scheduling (a) without Re-timing (b) R-DAG (c) R-CTG

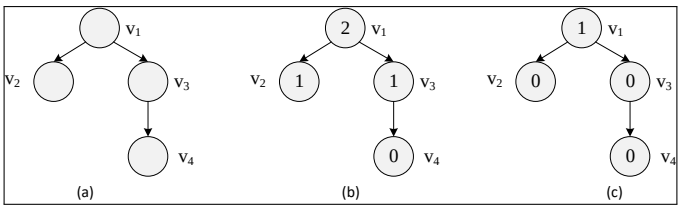


Fig. (6) Periodic Application (a) DAG (b) R-DAG (c) R-CTG

Although re-timing is effective in reducing energy consumption, there is a cost associated with it. One drawback of re-timing is that it adds prologue. The prologue latency is the number of periods in prologue times the period, T. The number of periods in the prologue is equal to the maximum re-timing value RT_{max} of the nodes in G, $RT_{max} = \max\{RT(v_i) : \forall v_i \in G\}$. Thus the prologue latency is:

$$prologLatency = RT_{max} \times T \tag{2}$$

Besides energy reduction, we also want to minimize the $prologLatency$. Fig. (5)(b) shows the re-timed schedule generated by R-DAG. Compared to this the prologue latency of the re-timed schedule generated by our approach is half. We are able to reduce the prologue latency because we take a different approach compared to R-DAG. We first transform the CTG into an independent task set by relaxing the precedence constraints between the nodes and then we schedule the independent task model onto the MPSoC. Hence, the MPSoC resources are maximally utilized and do not remain idle due to precedence constraints between nodes. Finally we calculate the re-timing values of the nodes and generate the re-timed schedule.

TABLE (II) Energy Consumption and Execution Time of Tasks of CTG in Fig. (6)

Task	pe1		pe2	
	Energy (nJ)	Execution Time	Energy (nJ)	Execution Time
V_1	2	1	1.5	1.5
V_2	2	1	1.5	1.5
V_3	2	1	1.5	1.5
V_4	2	1	1.5	1.5

Algorithm 1 describes our schedule-aware software pipelining scheduling approach. Our approach has three main steps.

1) **Step 1 (Line 1):** Use Algorithm 2 to generate the relaxed schedule π and task to processor mapping map.
2) **Step 2 (Line 2):** Given the CTG $G(v, E, A, W, X)$ and task to processor mapping map transform G into an extended graph G_e by adding additional nodes for every directed edge in G whose tail and head nodes are mapped on a different processor. An extended graph G_e is a directed acyclic graph $G(V + V^*, E)$. V is the set of original nodes that are kept unchanged and are called task nodes. For every edge $(v_i, v_j) \in E$ we insert an additional communication node v_c in V^* if v_i and v_j are mapped on different processors and replace (v_i, v_j) by two directed directed edges (v_i, v_c) and (v_c, v_j).
3) **Step 3 (Lines 4-9):** Calculate the re-timing values of the nodes. Given a node v_i and its child node v_j, our re-timing function is defined as follows:

$$RT(v_i) = \begin{cases} 0 & \text{if } v_i \text{ is sink node} \\ \max\{RT(v_i), RT(v_j) + 1\} & \text{if } \zeta(v_i) < \rho(v_j) \\ \max\{RT(v_i), RT(v_j)\} & \text{otherwise} \end{cases} \tag{3}$$

A. ALI-EBAD, A Relaxed Offline Scheduling and Voltage Scaling Algorithm

In this section, we describe our relaxed offline scheduling and voltage scaling approach. A relaxed schedule is generated by assuming that the precedence constraints between the nodes of a CTG do not exist, that is the nodes are assumed to be

Algorithm 1: R-CTG

input : A CTG G, tasks Deadlines, an MPSoC
output: Re-timed schedule

1 Use Algorithm 2 to generate the relaxed schedule π and task to processor mapping map;
2 Given the CTG G and task to processor mapping map transform G into an extended graph G_e;
3 Set the re-timing values of all leaf nodes of G_e to zero;
4 **for** ***each*** *node v_i in the reverse topological order of G_e* **do**
5 **for** ***each*** *parent v_j of v_i* **do**
6 **if** $\rho(v_i) < \zeta(v_j)$ **then**
7 $RT(v_i) \leftarrow \max\{RT(v_i), RT(v_j) + 1\}$;
8 **else**
9 $RT(v_i) \leftarrow \max\{RT(v_i), RT(v_j)\}$;
10 Given the relaxed schedule π and retiming values, generate the re-timed schedule.

Algorithm 2: ALI-EBAD

input : CTG G, matrix NC, set X, and NoC based MPSoC
output: Schedule π^{best} and an array Map reflecting task mapping

1 Compute the successor-tree-consistent deadline of all the nodes in G;
2 Create a list R and insert in it all the nodes of G, $R \leftarrow V$;
3 Create and array Map of size $|V|$;
4 Create two empty sets V_s and V_s^* ;
5 **repeat**
6 Set E_{exp}^{best} to ∞;
7 **for** ***each*** $v_l \in R$ **do**
8 **for** ***each*** $pe_k \in P$ **do**
9 Tentatively map v_l to pe_k;
10 Insert v_l in V_s;
11 **for** ***each*** *parent of v_l mapped on a different processor* **do**
12 Insert a communication node v_c in V_s^*
13 **for** ***each*** *child of v_l mapped on a different processor* **do**
14 Insert a communication node v_c in V_s^*
15 Solve the NLP to generate the schedule π and calculate the expected energy consumption E_{exp};
16 Round the voltage of each island that has been assigned invalid voltage level by NLP to a nearest highest valid voltage level and recompute the E_{exp} under new voltage settings.;
17 **if** $E_{exp} < E_{exp}^{best}$ **then**
18 Set E_{exp}^{best} to E_{exp};
19 Set π^{best} to π ;
20 Set j to k;
21 Set i to l;
22 Delete v_l from V_s and corresponding communication nodes from V_s^*;
23 Set $Map[i]$ to j;
24 Insert v_i in V_s;
25 **for** ***each*** *parent of v_i mapped on a different processor* **do**
26 Insert a communication node v_c in V_s^*
27 **for** ***each*** *child of v_i mapped on a different processor* **do**
28 Insert a communication node v_c in V_s^*
29 Delete v_i from R;
30 **until** *R is not empty*;

independent. We propose an offline scheduler Algorithm 2, $ALI-EBAD$ to generate the relax schedule.

Given a periodic CTG that models a streaming application, our main objective is to execute the application on heterogeneous VFI-NoC-MPSoC such that the total expected energy consumption is minimized. Unlike other state-of-the-art algorithms for task scheduling, we consider energy performance profiles, scheduling and voltage scaling in a integrated manner in the design of $ALI-EBAD$.

$ALI-EBAD$ maintains a ready list R that contains all the ready nodes and source nodes. A task is ready if all its parents have been scheduled. All the nodes of G are in R because we relax the precedence constraints between the task (Line 2). Next $ALI-EBAD$ repeats executing the following five steps until all the tasks in R have been scheduled.

1) Selects one by one each ready task $v_i \in R$, tentatively map v_i on each processor $pe_k \in P$. For each task and processor pair (v_l, pe_k), repeat the following:
   a) Insert task v_l in set V_s (Line 10). The set V_s contains all the tasks that have been scheduled. For each parent and child node of v_l mapped on a different processor insert a communication node in V_s^* (Line 11-14). The communication node is required because the precedence constraints have only been relaxed not removed. The data has to be transmitted over the NoC from the processor, where the parent node is mapped to the processor where child node is mapped.
   b) Solve the NLP described in the next section to generate the schedule π and calculate the expected energy consumption of G_s (Line 15). The schedule π specifies the unique start time, finish time for each node in G_s and a voltage setting for each island.
   c) Round the voltage of each island that has been assigned an invalid voltage level by NLP to the nearest highest valid voltage level (Line 16). Note that the schedule needs to be updated in this case. This involves re-calculating the start and finish times of task and communication nodes under new voltages settings such that the relative order of the task and communication nodes remain the same.
   d) Delete v_l from V_s (Line 22) and the corresponding communication nodes from V_s^*. v_l is deleted from V_s because its current mapping is tentative.

2) Find the task processor pair (v_i, pe_k), such that mapping v_i to pe_k results in a minimum increase in energy consumption amongst all the pairs.
3) Map v_i to pe_k and insert v_i to V_s (Lines 23-24).
4) For each parent and child node of v_i mapped on a different processor insert a communication node in V_s^* (Lines 25-28).
5) Delete v_i form R (Line 29).

1) NLP Based DVFS Approach: We propose an NLP based offline scheduler that is inspired by the scheduler proposed in [16], [46]. Before we describe our NLP based offline scheduler, we discuss the priority scheme used by the scheduler. Our approach uses the priority scheme earliest successor-tree-consistent deadline first [35] because it allows the DVFS scheme to efficiently utilize the available slack and significantly reduce energy consumption. The successor-tree-consistent deadline is defined as the upper bound on the latest finish time of the nodes in CTG. Compared to edge-consistent deadline it is a tighter bound on latest finish time of the nodes, because it takes into account the resource constraints of the MPSoC while calculating the latest finish time. Our NLP based offline approach, schedules tasks and communication nodes in the earliest successor-tree-consistent deadline manner, which means that nodes with shorter successor-tree-consistent deadline are scheduled earlier than nodes with longer successor-tree-consistent deadline.

Next, we describe our NLP based offline scheduler:

Operating frequency constraints: The operating frequency f_j of each island $c_j \in C$ is determined by the following constraints [47]:

$$f_j = \frac{((1+K_1)V_{dd_j} + K_2 V_{bs} - V_{th_1})^\alpha}{K_6 L_d V_{dd_j}} \quad \forall c_j \in C \tag{4}$$

where K_1, K_2, K_6 and V_{th_1} are circuit dependent constants, L_d is the logic depth, and α $(1.4 \leq \alpha \leq 2)$ is velocity saturation imposed by the technology used.

Execution and transmission time constraints: The execution time of each task node $v_i \in V$ is given by the following equation:

$$\Delta t_i = \frac{NCC_{i,k}}{f_j} \quad \forall v_i \in V_s \tag{5}$$

where v_i is mapped on processor pe_k that belongs to island c_j whose frequency is f_j.

Consider a communication node v_j whose parent task node v_p is mapped on pe_{src} and child task node v_c is mapped on pe_{dest}, the routing algorithm used by the network generates the route R_j from pe_{src} to pe_{dst}. The route $R_j =< L_1, L_2, \ldots, L_l >$ is an ordered list of links, where L_1 is the first link and L_l is the last link on the route.

For communication nodes we only consider the link transmission time and ignore the overheads such as inter router delay, data copy between buffers etc. The transmission frequency of a link $L_\gamma \in R_j$ is the minimum of the sender and receiver routers frequencies. Hence, the transmission time of a communication node v_j on link $L_\gamma \in R_j$ is given by the following constraint:

$$\Delta t_j(L_\gamma) = \frac{\chi_{p,c}}{b_w \lambda} \tag{6}$$

λ is given by the following constraint:

$$\lambda = \min\{f_u, f_v\} \tag{7}$$

where f_u and f_v are the frequencies of sender and receiver routers respectively.

Link causality constraints: In communication scheduling, network resources such as links are treated as processors in a way that each communication can only use one resource at a time. Hence, communication nodes are scheduled on the links for the time they occupy them.

Note that the route depends only on the source and destination of the communication because in our network model we assume deterministic (XY) routing. Furthermore, the entire communication must be transmitted on the established route because in the network model we suppose circuit switching. A communication node utilizing this route must be scheduled on all the links (of this route). The data traverses these links in the order they appear in the route vector.

The schedule of each communication node $v_j \in V^*$ on the links of the route $R_j =< L_1, L_2, \ldots, L_l >$ (that v_j traverses) must obey the link causality constraints according to cut-through switching [48], [38]. The link causality constraints are defined as follows:

$$\rho(v_j, L_1) \leq \rho(v_j, L_\gamma) \tag{8}$$

$$\rho(v_j, L_{\gamma-1}) + \Delta t_j(L_{\gamma-1}) \leq \rho(v_j, L_\gamma) + \Delta t_j(L_\gamma) \quad for\ 1 < \gamma \leq l \tag{9}$$

Resource exclusiveness constraints: In a feasible schedule nodes mapped on the same resource must not overlap. However, a schedule is deemed feasible even though mutually exclusive nodes may schedule in the same time interval. This is because only one among the mutually exclusive nodes execute at run-time thus, resource exclusiveness constraints are not violated.

We define resource exclusiveness constraints to order concurrent nodes mapped on the same resource in an exclusive manner. For each task node $v_i \in V$ the resource exclusiveness constraints are defined as follows:

$$\rho(v_i) + \Delta t_i \leq \rho(v_j) \quad \forall v_j \in \Pi(v_i, pe_k) \tag{10}$$

where $\Pi(v_i, pe_k)$ is a set of task nodes concurrent to v_i, have shorter or equal successor-tree-consistent deadline than v_i and are mapped on pe_k. Nodes v_i and v_j are concurrent if they are not reachable from each other in the CTG and are not mutually exclusive.

Similarly for each communication node $v_i \in V^*$ where the route that v_j takes is $R_j =< L_1, L_2, \ldots, L_l >$, the resource constraints are defined as follows:

$$\rho(v_i, L_\gamma) + \Delta t_i(L_\gamma) \leq \rho(v_j, L_\gamma) \quad \forall v_j \in \Pi(v_i, L_\gamma) \quad for\ 1 < \gamma \leq l \tag{11}$$

where $\Pi(v_i, L_\gamma)$ is a set of communication nodes concurrent to v_i, have shorter or equal successor-tree-consistent deadline than v_i and use the same L_γ.

Deadline constraints: We define the deadline constraints so that tasks complete execution before their deadlines as follows:

$$\rho(v_i) + \Delta t_i \leq d_i \quad \forall v_i \in V \tag{12}$$

Supply voltage and start time bounds: Given the minimum supply voltage V_{dd}^{min} and the maximum supply voltage V_{dd}^{max} the following constraints define the upper and lower bounds on the supply voltage assigned to each island:

$$V_{dd}^{min} \leq Vdd_j \leq V_{dd}^{max} \quad \forall c_j \in C \tag{13}$$

The following constraint define the lower bound on the start time of each task node $v_i \in V$:

$$\rho(v_i) \geq 0 \quad \forall v_i \in V \tag{14}$$

Objective function: The objective of our NLP formulation is to minimize the total expected energy consumption

$$\text{minimize} \quad E$$

where the total expected energy is given as follows:

$$E = \sum_{\forall v_i \in V} p(v_i)E_i + \sum_{\forall v_u \in V^*} p(v_u)E_u \tag{15}$$

E_i is the energy consumed in execution of a task v_i mapped on processor pe_k that belongs to VFI c_j:

$$E_i(V_{dd_j}) = (C_{eff_k} V_{dd_j}^2 NC_{i,k} + L_g(V_{dd_j} K_3 e^{K_4 V_{dd_j}} e^{K_5 V_{bs}} + |V_{bs}|I_{jn})\Delta t_i \tag{16}$$

where C_{eff_k} is the effective switched capacitance of pe_k, L_g denotes the number of logic gates, $\{K_3, K_4, K_5\}$ represents technology specific parameters, v_{bs} and I_{jn} represent body-bias voltage and leakage current respectively. E_u is the energy consumed in the execution of communication node v_u that traverses the route $R_u =< L_1, L_2, \ldots, L_l >$. The parent and child task nodes of v_u are v_p and v_c respectively. E_u is calculated as follows:

$$E_u = \sum_{\beta=1}^{l-1} \chi_{p,c} E_{bit}(\beta, \beta+1) \tag{17}$$

where $E_{bit}(src, dest)$ is the energy consumed to transmit one bit from src tile to dest. $E_{bit}(src, dest)$ is discussed in detail in [49].

V. Performance Evaluation

In this section we explain our experimental setup used for producing various results on different real benchmarks to compare the performance of our scheduler.

TABLE (III) Operating Frequency and Power Consumption of Type 1 and Type 2 Processors

Type 1 (Cortex A15)							
Frequency (GHz)	2.0	1.8	1.6	1.4	1.2	1.0	0.8
Power (mW)	2500	1750	1350	1000	850	650	400
Type 2 (Cortex A7)							
Frequency (GHz)	1.4	1.2	1	0.8	0.6	0.4	0.2
Power (mW)	82.0	76.0	74.0	72.0	68.0	66.0	64.0

TABLE (IV) The 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

A. Experimental Setup

We use Samsung Exynos 5422 chip energy model adapted from [50], in simulations we deploy two types of processors. Type 1 is a high-performance and high-energy consuming Cortex A15 (big). Type 2 is a low-power, low-power consuming Cortex A7 (little). The Cortex A15 consumes $\sim 6-12$ times higher power compared to Cortex A7 [51]. The operating frequencies and relative power consumption of both types are listed in TABLE (III). We adopted 70 manometers (nm) processor technology parameters from Ali et al. [9] given in TABLE (IV). We built the simulation environment in Matlab version R2016a. We use Matlab's fmincon function to solve the NLP problem.Moreover, we conducted the experiments using the hardware platform of Intel (R) Xeon (R), i5-3570 CPU with the clock frequency of 3.50 GHz and 16.00 GB memory, 10 MB cache.

We perform experiments on 12 real benchmarks listed in TABLE (V) and TABLE (VI). In TABLE (V), the robot benchmark contains tasks for automation and control. ATR is a real-time streaming application used for pattern recognition. Consumer-1 and consumer-2 consist of tasks to perform RGB to CMYK conversion and JPEG compression/decompression. The mp3-decoder benchmark performs Huffman Decoding (HD) and Inverse Discrete Transform (IDCT). Office benchmark contains tasks for text processing, image rotation, and gray-scale to binary conversion. In TABLE (VI), E shows the number of edges and O represents the number of OR-Fork nodes while C denotes the number of conditions for each benchmark. Cruise-control (ctg-1) and mjpeg-decoder (ctg-2) benchmarks represent the vehicle cruise controller system application and Motion-JPEG decoder respectively. The last four benchmarks symbolize synthetic benchmarks with conditional precedence constraints.

We use contention and energy-aware task mapping and edge scheduling (CA-TMES) approach developed by Han et al. [24] as a baseline for energy-efficiency performance compari-

TABLE (V) Characteristics of Benchmarks Without Conditional Precedence Constraints

Benchmark	No.of Nodes	No.of Edges
Robot	88	131
atr	14	15
consumer-1	7	8
consumer-2	5	4
mp3-decoder	17	18
office	3	2

TABLE (VI) Characteristics of Benchmarks With Conditional Precedence Constraints

Benchmark	No.of Nodes	E/O/C
cruise-control (ctg-1)	32	35/2/4
mpjeg-decoder (ctg-2)	14	13/0/0
ctg-3	28	26/1/3
ctg-4	29	27/2/4
ctg-5	25	23/3/8
ctg-6	34	36/3/7

son of our energy management technique, ALI-EBAD. The authors presented CA-TMES-Search and CA-TMES-Quick two different scheduling techniques for processor selection on which a task can start earliest among all other processors. CA-TMES-Search approximates the start time for each task while considering the communication contention. CA-TMES-Quick first maps the tasks and then determines the routes for communications. CA-TMES-Search relatively saves higher energy than CA-TMES-Quick because of coordinating the task mapping in an exhaustive way and subsequently, reducing the overall makespan significantly. Similarly, we compare the effectiveness of our re-timing technique R-CTG with a state-of-the-art approach called R-DAG developed by Wang et al. [23]. R-DAG transforms a set of periodic dependent tasks into periodic independent tasks. R-DAG technique assigns a re-timing value to each node in the DAG starting from reverse topological order. It assigns a value 0 to the sink node while adds 1 with the re-timing value if it is a parent node.

B. Results and Discussion

In this section, we perform experiments on a set of benchmarks for two different scenarios (1) without and (2) with conditional precedence constraints. We consider that the number of VFI (NVFI) = 4 and number of processors per VFI (NPI) = 4. We use two terms in the results i.e. heterogeneous VFI based NoC-MPSoC (VFI-NoC-HMPSoC) and homogeneous VFI based NoC-MPSoC (VFI-NoC-MPSoC).

1) Without Re-timing: Fig. (7) shows the energy consumption comparison of our task scheduling, ALI-EBAD with state-of-the-art energy management schemes CA-TMES-Search and CA-TMES-Quick. The horizontal axis represents real benchmarks while the vertical axis denotes the energy consumption in milli joules (mJ). ALI-EBAD outperforms CA-TMES-Search and CA-TMES-Quick scheduling techniques in terms of energy-efficiency. It achieves an average energy savings of $\sim 15\%$, and $\sim 20\%$ over CA-TMES-Search and CA-TMES-Quick respectively when homogeneous processors only of type 1 are used to form VFI-NoC-MPSoC architecture. Unlike, CA-TMES-Search and CA-TMES-Quick, ALI-EBAD performs task mapping, ordering and voltage scaling in an integrated manner. Moreover, it schedules dependent tasks closer to each other to avoid energy dissipation due to the utilization of links, buffers, and switches for communications. In case of task scheduling using both type 1 and type 2 processors to form VFI-NoC-HMPSoC. We randomly select the type of processor for each VFI in VFI-NoC-HMPSoC architecture to generate a heterogeneous computing platform while ensuring unbiased experimentation. The energy-efficiency further increases to $\sim 20\%$ and $\sim 25\%$ compared to CA-TMES-Search and CA-TMES-Quick when VFI-NoC-HMPSoC computing architecture is considered during task scheduling. This further reduction in energy consumption occurs because ALI-EBAD maps higher energy consuming tasks on high energy-efficient and low-performance processor. In other words it considers the energy performance profiles of the processors during task scheduling.

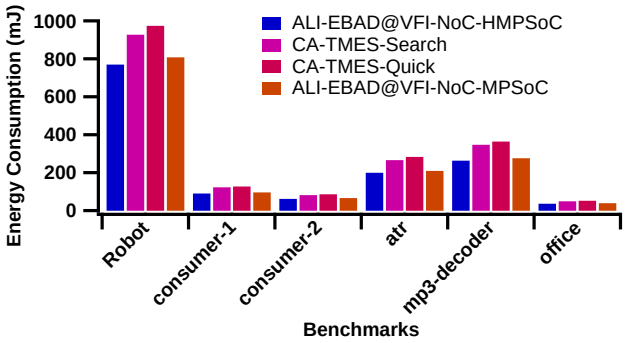


Fig. (7) Unconditional Benchmarks Energy Consumption without Re-timing

2) With Re-timing: Fig. (8) demonstrates the energy consumption of real benchmarks without conditional precedence when re-timing is deployed. We combine ALI-EBAD static task scheduler with our developed re-timing technique R-CTG while we integrate CA-TMES-Search and CA-TMES-Quick with R-DAG. R-CTG coarse-grained software pipelining transforms the intra-period dependencies into inter-period dependencies to utilize the wasted slack and efficiently utilize the DVFS for achieving higher energy-efficiency. The energy consumption significantly reduces when re-timing technique is used. Compared to CA-TMES-Search in Fig. (7) without re-timing the energy-efficiency increases to an average $\sim 40\%$ and $\sim 45\%$ for ALI-EBAD@VFI-NoC-MPSoC and ALI-EBAD@VFI-NoC-HMPSoC. Similarly Fig. (9) illustrates the energy consumption of benchmarks with conditional precedence constraints. It is noticeable that both R-DAG and R-CTG perform similarly in terms of energy-efficiency when combined with ALI-EBAD heuristic for both homogeneous and heterogeneous VFI-NoC-MPSoC platforms. Though, there is no significant energy performance improvement of R-CTG over R-DAG however, it reduces the maximum re-timing (RT_{max}) values significantly as shown in Fig. (10). R-CTG reduces the RT_{max} by 50% when compared to R-DAG. We compare the prologue latency in terms of maximum re-timing RT_{max}. The smaller the value of RT_{max} the shorter the

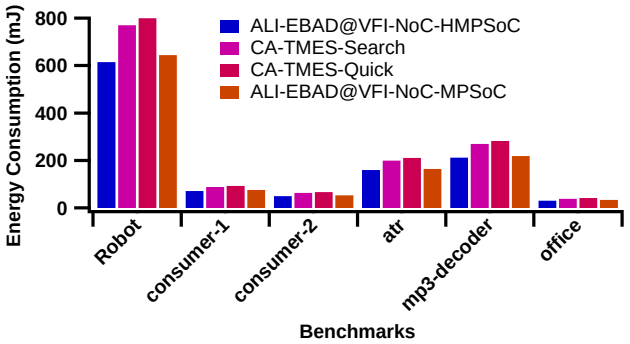


Fig. (8) Unconditional Benchmarks Energy Consumption with Re-timing

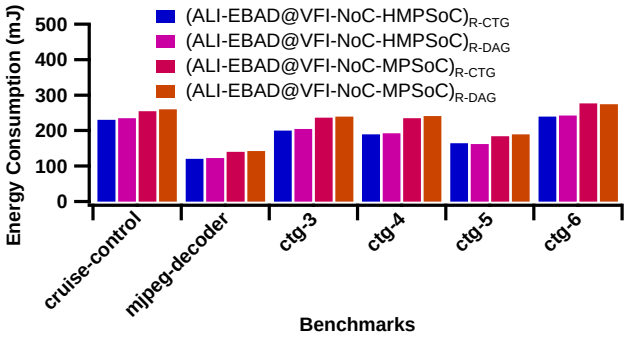


Fig. (9) Conditional Benchmarks Energy Consumption with Re-timing

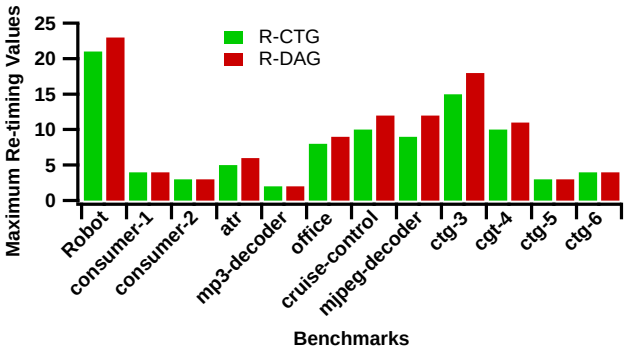


Fig. (10) Benchmarks Maximum Re-timing Values

prologue latency. Unlike R-DAG our novel re-timing, R-CTG achieves lower prologue because it only re-times tasks that free up the wasted-slack.

Concisely, ALI-EBAD static task scheduler using VFI-NoC-HMPSoC outperforms CA-TMES-Search and CA-TMES-Quick. It achieves an average energy-efficiency of $\sim 20\%$ over CA-TMES-Search and $\sim 25\%$ over CA-TMES-Quick. This energy saving increases to $\sim 40\%$ and $\sim 45\%$ when re-timing is deployed. R-CTG and R-DAG achieve similar energy-efficiency when integrated with ALI-EBAD but R-CTG produces a lower prologue of 50% compared to R-DAG.

VI. Conclusion

The computational complexity of real-time multimedia applications is rapidly proliferating, Voltage Frequency Islands (VFI) based Multiprocessor System-on-Chip (MPSoC) architectures are adopted for higher performance and effective energy management. In this paper we investigated complex scheduling problem for tasks, both with and without conditional precedence constraints by deploying a VFI-NoC-MPSoC computing platform. We proposed a novel re-timing technique, R-CTG and integrated it with a non linear programming based scheduling and voltage scaling approach referred to as ALI-EBAD. The R-CTG minimizes the latency caused by re-timing without compromising on the energy-efficiency. It significantly reduces the re-timing latency because it only re-times tasks that free up the wasted slack. We conducted an experiment on 12 benchmarks the results of which demonstrate that ALI-EBAD deploying VFI-NoC-HMPSoC outperforms CA-TMES-Search and CATMES-Quick while achieving an average energy-efficiency improvement of $\sim 20\%$ and $\sim 25\%$ respectively. The energy saving significantly increases to $\sim 40\%$ and $\sim 45\%$ when R-CTG is used. Compared to a previous state-of-the-art re-timing technique, R-DAG, our coarse-grained software pipelining, R-CTG achieves similar energy-efficiency when integrated with ALI-EBAD however it improves computational efficiency by reducing prologue by $\sim 50\%$. In the future we can also consider Quality-of-Experience (QoE), which is an interesting parameter from a user's perspective.

References

[1] A. P. Plageras, K. E. Psannis, Y. Ishibashi, and B.-G. Kim, "Iot-based surveillance system for ubiquitous healthcare," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2016, pp. 6226–6230.

[2] H. Ali, U. U. Tariq, M. Hussain, L. Lu, J. Panneerselvam, and X. Zhai, "Arsh-fati a novel metaheuristic for cluster head selection in wireless sensor networks," *IEEE Systems Journal*, 2020.

[3] J. Chase, "The evolution of the internet of things," *Texas Instruments*, vol. 1, pp. 1–7, 2013.

[4] A. Aliyu, A. H. Abdullah, O. Kaiwartya, Y. Cao, J. Lloret, N. Aslam, and U. M. Joda, "Towards video streaming in iot environments: Vehicular communication perspective," *Computer Communications*, vol. 118, pp. 93–119, 2018.

[5] Y. Zhang, L. Sun, H. Song, and X. Cao, "Ubiquitous wsn for healthcare: Recent advances and future prospects," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 311–318, 2014.

[6] P. Verma and S. K. Sood, "Fog assisted-iot enabled patient health monitoring in smart homes," *IEEE Internet of Things Journal*, 2018.

[7] H. Ali, U. U. Tariq, L. Liu, J. Panneerselvam, and X. Zhai, "Energy optimization of streaming applications in iot on noc based heterogeneous mpsocs using re-timing and dvfs," in *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 2019, pp. 1297–1304.

[8] K. Huang, X. Zhang, D. Zheng, M. Yu, X. Jiang, X. Yan, L. B. de Brisolara, and A. A. Jerraya, "A scalable and adaptable ilp-based approach for task mapping on mpsoc considering load balance and communication optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2018.

[9] H. Ali, U. U. Tariq, Y. Zheng, X. Zhai, and L. Liu, "Contention & energy-aware real-time task mapping on noc based heterogeneous mpsocs," *IEEE Access*, vol. 6, pp. 75 110–75 123, 2018.

[10] M. Ahmadi, W. J. Gross, and S. Kadoury, "A real-time remote video streaming platform for ultrasound imaging," in *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*. IEEE, 2016, pp. 4383–4386.

[11] A. Kurth, A. Tretter, P. A. Hager, S. Sanabria, O. Göksel, L. Thiele, and L. Benini, "Mobile ultrasound imaging on heterogeneous multi-core platforms," in *Proceedings of the 14th ACM/IEEE Symposium on Embedded Systems for Real-Time Multimedia*. ACM, 2016, pp. 9–18.

[12] H. Archana and V. P. KS, "An investigation towards effectiveness in image enhancement process in mpsoc," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 2, pp. 963–970, 2018.

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

[14] H. Ali, X. Zhai, U. U. Tariq, and L. Liu, "Energy efficient heuristic algorithm for task mapping on shared-memory heterogeneous mpsocs," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2018, pp. 1099–1104.

[15] H. Ali, U. U. Tariq, X. Zhai, and L. Liu, "Energy efficient task mapping & scheduling on heterogeneous noc-mpsocs in iot based smart city," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2018, pp. 1305–1313.

[16] U. U. Tariq and H. Wu, "Energy-aware scheduling of conditional task graphs with deadlines on mpsocs," in *Computer Design (ICCD), 2016 IEEE 34th International Conference on*. IEEE, 2016, pp. 265–272.

[17] ——, "Energy-aware scheduling of periodic conditional task graphs on mpsocs," in *Proceedings of the 18th International Conference on Distributed Computing and Networking*. ACM, 2017, p. 13.

[18] U. Ullah Tariq, H. Ali, L. Liu, J. Panneerselvam, and X. Zhai, "Energy-efficient static task scheduling on vfi based noc-hmpsocs for intelligent edge devices in cyber-physical systems," *ACM Transactions on Intelligent Systems and Technology*, 2019.

[19] D. E. Lackey, P. S. Zuchowski, T. R. Bednar, D. W. Stout, S. W. Gould, and J. M. Cohn, "Managing power and performance for system-on-chip designs using voltage islands," in *Computer Aided Design, 2002. ICCAD 2002. IEEE/ACM International Conference on*. IEEE, 2002, pp. 195–202.

[20] E. L. de Souza Carvalho, N. L. V. Calazans, and F. G. Moraes, "Dynamic task mapping for mpsocs," *IEEE Design & Test of Computers*, vol. 27, no. 5, pp. 26–35, 2010.

[21] U. U. Tariq, H. Ali, L. Liu, and X. Zhai, "A novel meta-heuristic for green computing on vfi-noc-hmpsocs," in *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 2019, pp. 1545–1552.

[22] S. Z. Sheikh and M. A. Pasha, "Energy-efficient multicore scheduling for hard real-time systems: A survey," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 17, no. 6, p. 94, 2018.

[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 Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 16, no. 2, p. 14, 2011.

[24] J.-J. Han, M. Lin, D. Zhu, and L. T. Yang, "Contention-aware energy management scheme for noc-based multicore real-time systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 3, pp. 691–701, 2014.

[25] S. Olafsson, "A general model for task distribution on an open heterogenous processor system," *IEEE transactions on systems, man, and cybernetics*, vol. 25, no. 1, pp. 43–58, 1995.

[26] H. Aydin, R. Melhem, D. Mossé, and P. Mejía-Alvarez, "Determining optimal processor speeds for periodic real-time tasks with different power characteristics," in *Real-Time Systems, 13th Euromicro Conference on, 2001*. IEEE, 2001, pp. 225–232.

[27] S. Tosun, "Energy-and reliability-aware task scheduling onto heterogeneous mpsoc architectures," *The Journal of Supercomputing*, vol. 62, no. 1, pp. 265–289, 2012.

[28] N. Kumar and D. P. Vidyarthi, "A ga based energy aware scheduler for dvfs enabled multicore systems," *Computing*, vol. 99, no. 10, pp. 955–977, 2017.

[29] P. Dziurzanski and A. K. Singh, "Feedback-based admission control for firm real-time task allocation with dynamic voltage and frequency scaling," *Computers*, vol. 7, no. 2, p. 26, 2018.

[30] 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 *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2010 16th IEEE*. IEEE, 2010, pp. 195–204.

[31] 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.

[32] D. Shin and J. Kim, "Power-aware scheduling of conditional task graphs in real-time multiprocessor systems," in *Proceedings of the 2003 international symposium on Low power electronics and design*. ACM, 2003, pp. 408–413.

[33] D. Wu, B. M. Al-Hashimi, and P. Eles, "Scheduling and mapping of conditional task graph for the synthesis of low power embedded systems," *IEE Proceedings-Computers and Digital Techniques*, vol. 150, no. 5, pp. 262–273, 2003.

[34] P. Eles, K. Kuchcinski, Z. Peng, A. Doboli, and P. Pop, "Scheduling of conditional process graphs for the synthesis of embedded systems," in *Design, Automation, and Test in Europe*. Springer, 2008, pp. 15–29.

[35] U. U. Tariq, H. Wu, and S. Abd Ishak, "Energy-aware scheduling of conditional task graphs on noc-based mpsocs," in *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.

[36] S. Pagani, J.-J. Chen, and M. Li, "Energy efficiency on multi-core architectures with multiple voltage islands," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 6, pp. 1608–1621, 2014.

[37] J. Liu and J. Guo, "Energy efficient scheduling of real-time tasks on multi-core processors with voltage islands," *Future Generation Computer Systems*, vol. 56, pp. 202–210, 2016.

[38] U. U. Tariq, H. Ali, L. Liu, J. Panneerselvam, and X. Zhai, "Energy-efficient static task scheduling on vfi-based noc-hmpsocs for intelligent edge devices in cyber-physical systems," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–22, 2019.

[39] A. Gammoudi, A. Benzina, M. Khalgui, and D. Chillet, "Energy-efficient scheduling of real-time tasks in reconfigurable homogeneous multicore platforms," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.

[40] Y. Wang, D. Liu, Z. Qin, and Z. Shao, "Optimally removing intercore communication overhead for streaming applications on mpsocs," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 336–350, 2011.

[41] Y. Wang, Z. Shao, H. C. 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 Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1797–1807, 2013.

[42] M. Lombardi, M. Milano, M. Ruggiero, and L. Benini, "Stochastic allocation and scheduling for conditional task graphs in multi-processor systems-on-chip," *Journal of scheduling*, vol. 13, no. 4, pp. 315–345, 2010.

[43] C. E. Leiserson and J. B. Saxe, "Retiming synchronous circuitry," *Algorithmica*, vol. 6, no. 1, pp. 5–35, 1991.

[44] Y. Wang, D. Liu, Z. Qin, and Z. Shao, "Optimally removing intercore communication overhead for streaming applications on mpsocs," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 336–350, 2013.

[45] Y. Wang, Z. Shao, H. C. 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 transactions on parallel and distributed systems*, vol. 25, no. 7, pp. 1797–1807, 2014.

[46] U. U. Tariq, H. Wu, and S. Abd Ishak, "Energy and memory-aware software pipelining streaming applications on noc-based mpsocs," *Future Generation Computer Systems*, 2020.

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

[48] O. Sinnen and L. A. Sousa, "Communication contention in task scheduling," *IEEE Transactions on parallel and distributed systems*, vol. 16, no. 6, pp. 503–515, 2005.

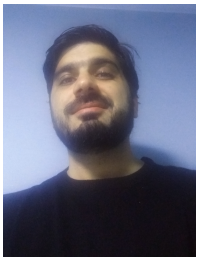
[49] U. Y. Ogras, R. Marculescu, D. Marculescu, and E. G. Jung, "Design and management of voltage-frequency island partitioned networks-on-chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 3, pp. 330–341, 2009.

[50] D. Liu, J. Spasic, G. Chen, and T. Stefanov, "Energy-efficient mapping of real-time streaming applications on cluster heterogeneous mpsocs," in

Embedded Systems For Real-time Multimedia (ESTIMedia), 2015 13th IEEE Symposium on. IEEE, 2015, pp. 1–10.

[51] A. Lukefahr, S. Padmanabha, R. Das, R. Dreslinski Jr, T. F. Wenisch, and S. Mahlke, "Heterogeneous microarchitectures trump voltage scaling for low-power cores," in *Proceedings of the 23rd international conference on Parallel architectures and compilation*. ACM, 2014, pp. 237–250.

Umair Ullah Tariq received his master degree from University of Engineering and Technology, Taxila, Pakistan and the PhD degree in computer science and engineering from University of New South Wales, Sydney, Australia. He is currently working on energy-aware task scheduling on multi-processor systems and intrusion detection system for IoT. He has published several research papers in prominent conferences and journals. His research interests include energy-aware task scheduling, digital image processing, computer vision and IoT security.



Haider Ali received master degree in electronic systems design engineering from Manchester Metropolitan University (MMU), U.K and PhD degree from the University of Derby (UoD), UK. He is currently a Lecturer at the Department of Electronics, Computing and Mathematics, UoD. He served COMSATS University Islamabad, Abbottabad Campus, Pakistan as a Lecturer from 2011 to 2016. 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, Internet-of-Things (IoT), algorithms design, and embedded systems. He has received 2 best paper awards from international conferences. He has served as member of the technical program committee for different workshops and serves many reputable journals as a reviewer.

Lu Liu is the Head of School of Informatics at the University of Leicester. Prior to this, he was the Head of School of Electronics, Computing and Mathematics and Professor of Distributed Computing at the University of Derby. Professor Liu received his PhD degree from Surrey Space Centre at the University of Surrey. He had worked as a Research Fellow at the WRG e-Science Centre at the University of Leeds. Professor Liu's research interests are in the areas of data analytics, AI, Cloud computing, service computing and the Internet of Things and he has over 200 scientific publications in reputable journals, academic books and international conferences. Professor Liu has secured and participated in many research projects which are supported by research councils, BIS, Innovate UK, British Council and leading industries. He received the Vice-Chancellor's Awards for Excellence in Doctoral Supervision in 2018, BCL Faculty Research Award in 2012 and was recognised as a Promising Researcher by the University of Derby in 2011. He has been the recipient of 7 Best Paper Awards from international conferences and was invited to deliver 7 keynote speeches at international conferences/workshops. Professor Liu is a Fellow of BCS (British Computer Society) and serves as an Editorial Board member of 6 international journals and the Guest Editor for 19 journal special issues. He has chaired over 30 international conference and workshops, and presently or formerly serves as the program committee member for over 60 international conferences and workshops.

James Hardy received his PhD degree in computer science at the University of Derby. He holds a B.Eng. in Electronic Engineering from Nottingham University and an M.Sc. in Computer Networks from the University of Derby complemented by professional networking qualifications. James has authored and co-authored papers on subjects including vehicular traffic control, virtualisation, IPv6 addressing, software ageing and Web Services ranking. Current research projects include Smart City Transportation, Connected and Autonomous Vehicles, Communication Systems, Green Computing, HaaS, IaaS, control systems, simulation and virtualisation.

Muhammad Kazim is working as a lecturer in Cyber Security at the De Montfort University, U.K. Earlier, he worked as a Research Software Engineer at the University of Derby. He holds a PhD degree in computer science from the University of Derby. He received a master's degree in computer security and a bachelor's degree in computer engineering from the National University of Sciences and Technology, Pakistan. Dr. Kazim has also worked at various academic positions at the University of Derby including, as an Associate Academic and a Graduate Teaching Assistant. His research interests include cloud security, IoT, edge computing, networks security, and distributed systems.

Waqar Ahmad received the BSc and MSc degrees in Computer Engineering from COMSATS University Abbottabad and University of Engineering and Technology Taxila, Pakistan, respectively. He did his PhD degree from the Department of Electronics and Telecommunications, Politecnico di Torino, Italy. He was a Higher Education Commission (HEC) fully funded PhD scholar. He is currently an Assistant Professor with University of Engineering and Technology, Pakistan. His current research interests focus on developing machine learning algorithms and architectures for video coding and multi-camera networks. He serves as reviewer for Journal of Circuits, Systems and Computers.