A Survey on System Level Energy Optimisation for MPSoCs in IoT and Consumer Electronics

Haider Ali[a], Umair Ullah Tariq[b], James Hardy[a], Xiaojun Zhai[c], Liu Lu[d], Yongjun Zheng[e], Faycal Bensaali[f], Abbes Amira[g], Kaniz Fatema[a] and Nikos Antonopoulos[h]

[a]*Department of Electronics, Computing and Mathematics, University of Derby, Derby, UK.*
[b]*School of Engineering and Technology, Central Queensland University, Sydney, Australia.*
[d]*School of Informatics, University of Leicester, Leicester, UK.*
[c] *School of Computer Science and Electronic Engineering at the University of Essex, Colchester, UK.*
[e] *School of Computing and Engineering at the University of West London, London, UK.*
[f] *College of Engineering, Qatar University, Doha, Qatar.*
[g]*School of Computer Science and Informatics, De Montfort University, Leicester, UK.*
[h]*Research and Innovation at Edinburgh Napier University, UK.*

ARTICLE INFO

Keywords:
Internet-of-Things
Smart City
WSN
SNs
Smart-phones
Bus
NoC
MPSoCs
Scheduling
DVFS
DPM
Re-timing
Energy-efficiency

ABSTRACT

Internet-of-Things (IoT) is an appealing service to revolutionise Smart City (SC) initiatives across the globe. IoT interconnects a plethora of digital devices known as Sensor Nodes (SNs) to the Internet. Due to their high performance and exceptional Quality-of-Service (QoS) Multiprocessor System-on-Chip (MPSoC) computing architectures are gaining increasing popularity for the computationally extensive workloads in both IoT and consumer electronics. In this survey, we have explored balance between the IoT paradigm and its applications in SC while introducing Wireless Sensor Network (WSN), including the structure of the SN. We considered MPSoCs systems in relation to characteristics such as architecture and the communication technology involved. This provides an insight into the benefits of coupling MPSoCs with IoT. This paper, also investigates prevalent software level energy optimisation techniques and extensively reviews workload mapping and scheduling approaches since 2001 until today for energy savings using (1) Dynamic Voltage and Frequency Scaling (DVFS) and/or Dynamic Power Management (DPM) (2) Inter-processor communication reduction (3) Coarse-grained software pipelining integrated with DVFS. This paper constructively summarises the findings of these approaches and algorithms identifying insightful directions to future research avenues.

1. Introduction

Internet-of-Things (IoT) is a technological communication revolution that bridges a plethora of modern digital devices, users and smart things to the Internet for numerous applications. Thus, IoT is transforming the Internet into a more pervasive and immersive model (1; 2). The literature demonstrates that the emergence of IoT has initiated Smart City (SC) concept, a paradigm that particularly concentrates on reconciling and enhancing both ecology and economy of city modernization (3; 4). An ultimate goal of IoT technology for the SC is to optimise and efficiently control the city systems. More precisely the fundamental aim is to boost the effectiveness of city governance by establishing a communicating link between the human users and smart technology. IoT is gaining popularity for smart cities in order to develop efficient and low-cost applications for purposes such as monitoring, control, automation etc.

In IoT based SC (IoT-SC) services information is collected from the users and/or smart things via wearable devices, sensors, and cameras (in Figure (1), these are represented by red, blue, and green colours respectively). The devices are commonly known as Sensor Nodes (SNs). Data gathered is transferred, processed and stored in the cloud for further post processing to enable visualisation and recom-

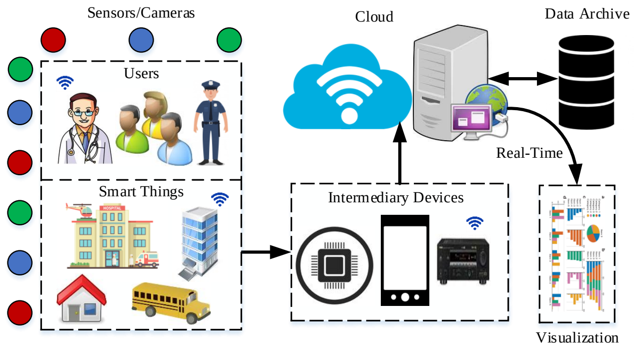

Figure (1): IoT connecting devices and/or users for various applications

mendations to be made, as shown in Figure (1) (5; 6). Italy is one of the first countries to offer large scale smart services (7). Table (1) shows a synoptic view of the Padova SC project deploying urban IoT in Padova city, Italy (8). Recently in China, a boom in IoT-SC initiatives occurred as listed in Table (2) to promote green, low-carbon and a sustainable development for 1.3 billion people (9).

The continuously expanding need for real-time applications has influenced the growth in the usage of Multiprocessor System-on-Chips (MPSoCs) in modern embedded systems (10). MPSoCs provide high performance, high-scale integration, exceptional Quality-of-Service (QoS), overwhelm-

h.ali@derby.ac.uk (H. Ali)
ORCID(s):

Table (1)
Padova smart city project (8)

Application	Network	Energy Source
Structural health monitoring	802.15.4; WiFi and Ethernet	Mostly battery powered
Waste management	WiFi; 3G and 4G	Energy harvester or battery powered
Air quality monitoring	802.15.4; Bluetooth and WiFi	Photovoltaic panels or battery powered
Noise monitoring	802.15.4 and Ethernet	Energy harvester or battery powered
Traffic congestion monitoring	802.15.4; WiFi and Bluetooth; Ethernet	Energy harvester or battery powered
City energy consumption	Ethernet	Main powered
Smart parking	802.15.4 and Ethernet	Energy harvester
Smart lighting	802.15.4; WiFi and Ethernet	Main powered
Home/buildings automation	802.15.4; WiFi and Ethernet	Mostly battery powered

Table (2)
Smart city services in China (9)

Urban Function	Smart Applications
Reproduction	Public safety, environment, energy, healthcare, household, and urban management.
Economic-Development	Manufacturing, industry, logistics, and city planning.
Social-Interactions	Public transportation, online shopping, and general social management.
Culture-Enjoyment	Education, tourism, and outdoor stream media.

ing reliability, and significant energy-efficiency in comparison to uniprocessor architectures (11). These qualities have contributed to the wide deployment of MPSoC as SNs for numerous real-time IoT applications. One of the major technological challenges for IoT is energy consumption optimisation at the SN level. High energy consumption of embedded systems at SN level not only reduces the lifetime of the IoT but also results in an increased carbon footprint. Approximately 4.7% of global electrical energy is consumed by ICT and results in the releases of 2.0% of overall atmosphere CO_2 (12). Moreover, SN mostly operates on embedded battery sources with limited residual energy, replacement of the batteries is usually challenging, difficult and expensive (13; 14).

Several of the survey papers reviewed considered different aspects of IoT technologies. Aztori et al. (15) presented communication technologies and reviewed different visions of IoT. Al-Fuqaha et al. (16) discussed protocols, enabling technologies, IoT applications and identified the open research problems. Andrea et al. (17) addressed security and privacy issues in IoT and presented security challenges and vulnerabilities. Botta et al. (18) highlighted the integration of IoT and Cloud while Jie et al. (19) reviewed fog/edge computing integration in IoT. Ray et al. (20) presented the architecture of the IoT and identified the research problems in the current trends of architecture. Shaikh et al. discussed efficient deployment of sensors, objects, and the Internet to reduce the energy consumption reduction in IoT (21). At the Wireless Sensor Network (WSN) level, Rault et al. (14) categorised the applications of WSN and discussed different energy conservation schemes. Kulkarni et al. (22) surveyed data-aggregation, clustering, and node localization in the WSN. Islam et al. (23) examined security and the reliability challenges of WSNs and surveyed their practicality for industrial adoption. Yetgin (24) presented a survey on the techniques to enhance the lifetime of WSN in IoT. At the SN level, Mittal (25) surveyed general energy-saving techniques for embedded systems. Siddiqui (26) discussed the security issues in IoT based MPSoC. Bambagini et al. (27) reviewed workload scheduling on multiprocessor systems considering Dynamic Voltage and Frequency Scaling (DVFS) and Dynamic Power Management (DPM) with no emphasis on coarse-grained software pipelining for dependent task scheduling. None of the surveys discussed the role of MPSoC in IoT plus consumer electronics and their various types, software level energy optimisation techniques, and energy savings approaches/algorithms. This review paper contributions are demonstrated in Figure (2).

The rest of the paper is organized as follows: we present the architecture of WSN and structure of SN in Section 2. In Section 3 we discuss MPSoCs applications in IoT. Section 4 explains the different types of MPSoC system. Section 5 overviews the relevant application and power models and discusses task scheduling. Section 6 presents the software level energy management techniques. Section 7 summarises the energy optimisation approaches and algorithms, Section 8 discusses the challenges and proposes some future research directions. Finally, we present the conclusions of our paper in Section 9.

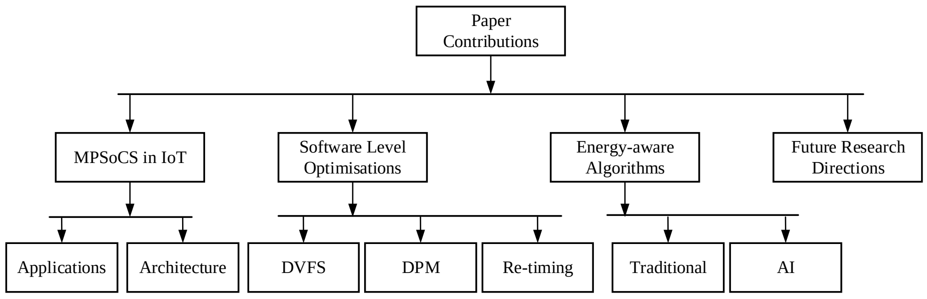

Figure (2): Review paper contributions

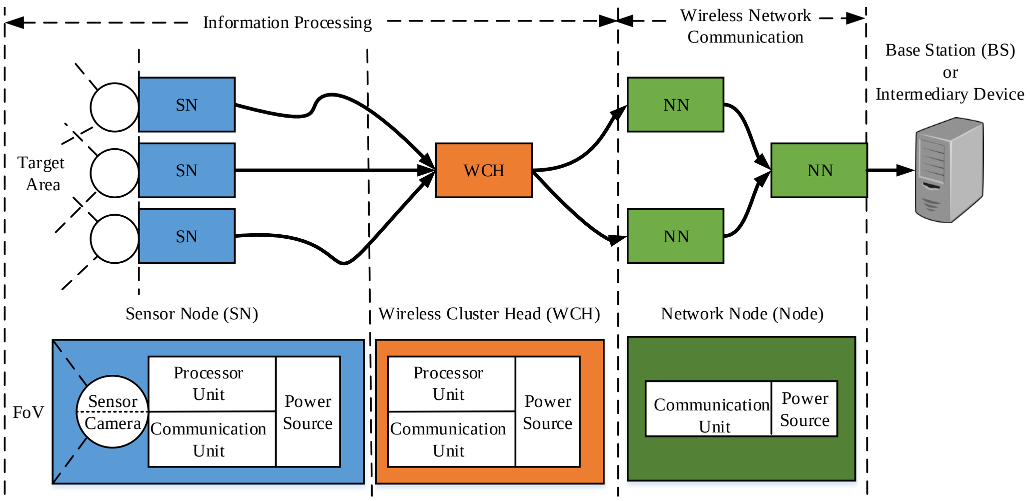

Figure (3): Wireless sensor network structure

2. Wireless Sensor Network and Sensor Nodes

The example WSN shown in Figure (3), is an integrated part of the IoT (28; 29). WSN gathers the information about the physical world through the sensors/cameras and store it in cloud. It is a vital resource necessary to implement the vision of IoT paradigm (30; 15; 31). in simple terms, WSN is network composed of resource-constrained digital devices known as SNs that are used to collect information from a Field-of-View (FoV) (32; 33). Advancements in modern technologies have had a significant positive impact on the availability of high-quality SN for numerous applications with superior technical features, low cost, minimal power consumption, and small physical size. The technological improvement in SN technology for multimedia data has enhanced the growth in WSN (34; 35). Multimedia content such as audio and video streams processing a promising technology utilised by numerous applications (36).

A typical SN (highlighted by blue colour in Figure (3)) is composed of four components (37). Each component is explained as follows:

1. *Sensing Unit:* Single or multiple sensors and Analogue to Digital Converters (ADC). The physical information is detected and captured in analogue form which the ADC converts to a digital format as required for the processing unit.
2. *Processing Unit:* Responsible for intelligent processing of the data received from the target area, this is a microprocessor and/or microcontroller with integrated memory. Digital Signal Processor (DSPr) and Application Specific Instruction-set Processor (ASIP) could also be component parts of the processing unit.
3. *Communication Unit:* Short-range transceiver system commonly based on standards such as IEEE 802.14.3, IEEE 802.15.4 or ZigBee$^{\text{TM}}$ although other protocols e.g. IEEE 802.1 can also be utilised for specialised applications such as Industrial IoT (IIoT).
4. *Power Source:* Regulated supply for the data collection, processing, and transmission subsystems. Batteries with limited residual energy are often deployed. The cells may be disposable primary or secondary if the additional complexity of energy collection is advantageous.

Ethernet is an established highly scalable network protocol with standard based compatibility and excellent bandwidth capability. However, it may not be a suitable choice for safety and real-time applications such as those required by IIoT and Industry 4.0. Other Ethernet extension protocols have also been suggested as alternatives e.g. TTEthernet, ARINC 664 Specification Part 7 (AFDX), and EtherCAT. Incompatibility between these protocols can however create problems and they cannot operate on the same physical links in the network without sacrificing real-time guarantees. Time-Sensitive Networking (TSN) provides a solution that consists of a set of standards, created by the IEEE 802.1 working group specifically for Industry 4.0. TSN categorizes flows into three main traffic-types depending upon the criticality (38).

1. Time-Triggered (TT) traffic tends to support hard real-time applications which require low latency and jitter. Concisely, TT traffic tends to support hard real-time applications which require low latency and jitter. TT traffic has the highest priority.
2. Audio-Video-Bridging (AVB) traffic is lower priority than TT and supports applications requiring bounded end-to-end latency.
3. Best-Effort (BE) traffic is the lowest priority. It is used for applications where timing guarantees are not required.

Worthy of note that one of the major challenges for Industry 4.0 and IIoT is the secure exchange of information between, machines, devices, and services across different industries. Therefore, Reference Architecture Model for Industries 4.0 (RAMI 4.0) recommends Open Platform Communications Unified Architecture (OPC UA) for implementing communication layer (39).

2.1. Sensor Node Architectures

In this section we discuss different architectures used in SNs for WSNs considering the energy-efficiency and performance requirements (40).

Due to the ubiquitous nature of the SNs in IoT, they need to be wirelessly connectable, capable of executing complex operations and energy-efficient. Some specific applications may impose other different additional constraints e.g. ruggedness, environment invisibility, and timing constraints. Thus, applications in IoT vary in terms of degree of interaction with the environment, communication need, and computation. However, energy-efficiency is a major requirement because SNs are mostly operated on battery and/or energy-harvesting devices. Applications are becoming more complex, smarter, and require immense processing power subsequently, energy-aware workload execution is a challenging task (41).

Recent advances in the micro-electro-mechanical systems technology, wireless communications, and digital electronics have enabled the development of high performance, low-cost, low-power, multi-functional SNs that are small in size and communicate untethered in short distances. They are the endpoints of the WSN and each SN comprises of sensor, processing unit and a communication unit (42; 37): Different architectures used as SNs in WSNs are discussed as follows:

2.1.1. Single-core Processors:

The transistor count of commercially available Intel processors has increased significantly since the original 4004 processor was released in 1971 (43; 44). The 4004 had 2300 transistors, by 2016 the number of transistors had increased by a factor of 3×10^6. Increasing the number of transistors increases capability but also increases the power requirements (45). Switching and leakage currents causes the temperature of the silicon die to increase. To prevent rapid and catastrophic chip failure, the heat is dissipated by increasing the surface area (and hence size) of the device. This is commonly combined with a forced cooling system. The power consumption attributed to switching is directly proportional to the switching speed and the number of transistors. Transistors operate at clock frequencies approaching 1 GHz while integrating more transistors on a chip for higher performance increasing the power dissipation. Given that for a fixed number of transistors, increasing clock frequency would the only way to improve performance for single-core, single processor systems, liquid cooling would be necessary to achieve modern day computation needs (43; 46).

2.1.2. Microcontroller:

A microcontroller consisting of antenna (for wireless communication), processor, memory, sensor, and DC battery has been widely used as SN in the WSNs. Atmel ATMega 128L, MSP430, and Mica2 are the popular examples used as SNs. Though the energy-efficiency of these microcontroller based SNs was phenomenal however, the processing capability was limited and there was a latency issue because of the slower response. Thus, new architectures have been developed to overcome the limitations of microcontrollers and one of the known example is Digital Signal Processors (DSPr).

2.1.3. Digital Signal Processors:

Some applications in WSNs may require to perform digital filtering, Fourier analysis, and encoding. Microcontrollers are generally not optimised for such operations therefore, DSPr are used to perform these data extensive mathematical operations. Specifically DSPr optimises handles and optimises digital signal processing related tasks. Though DSP is a suitable solution for many applications but there are limitations such as lower speed and bandwidth. However DSPr can be an integrated part of the MPSoCs though some studies suggested ASIC (Application Specific Integrated Circuit).

2.1.4. ASIC:

ASIC is an electronic circuit that integrates all components on a single chip required for performing a special tasks. ASIC has high performance, decreased circuit's congestion, and low power consumption. These qualities make them ideal for being deployed in WSNs. Though ASIC provide an energy-efficient and robust computing platform for data extensive applications in WSNs however, time-to-market, price, lack of scalability and flexibility are the disadvantages of ASIC. Thus Field Programmable Gate Arrays (FPGAs) have replaced ASICs as an alternative.

2.1.5. FPGA:

It is an integrated circuit that is designed to be configured by the customers or designers after manufacturing. FPGAs have higher adaptability and flexibility compared to ASIC.

The FPGA architectures are re-configurable but complex to design though, they are useful for complex applications in WSNs. Xilinx Virtex-4 is an example of FPGA based architecture used in WSNs for video processing related applications (compression, decoding, image processing). FPGAs have gained drastic acceptance in WSNs to fulfill the requirements such as performance and flexibility.

2.1.6. *Multiprocessor System-on-Chips (MPSoCs):*

Multiprocessor systems are beneficial for developing high performance and energy-efficient systems such as green computing. MPSoC is set of independent and interconnected processors integrated on a silicon chip. These processors cooperate and communicate with each other to execute applications (43; 46). MPSoC is a single chip system that integrates all or most of the functions of an electronic system including I/O units with analog and mixed-signal components, memory, instruction-set processors, buses, specialized logic, and digital signal processing functions (45; 47). MPSoC has set a new direction to the field of the embedded system. Modern MPSoC architectures also integrate Graphics Processor Unit (GPU), USB controller, Ethernet and/or wireless radios (3G, 4G, WiFi, 4G LTE), power management circuits, and multi-core functions (48; 49; 50). MPSoCs have been pioneered by CPU manufacturing companies such as Xilinx, Tilera, IBM, Motorola, Intel, Samsung, and Apple (51; 52).

MPSoC is a single chip system that integrates all or most of the functions of an electronic system including I/O units with analog and mixed-signal components, memory, instruction-set processors, buses, specialized logic, and digital signal processing functions (45; 47). MPSoC has set a new direction to the field of the embedded system. Modern MPSoC architectures also integrate Graphics Processor Unit (GPU), USB controller, Ethernet and/or wireless radios (3G, 4G, WiFi, 4G LTE), power management circuits, and multi-core functions (48; 49; 50). MPSoCs have been pioneered by CPU manufacturing companies such as Xilinx, Tilera, IBM, Motorola, Intel, Samsung, and Apple (51; 52). Suitable examples and architectures of the MPSoCs from these companies have been provided at different sections of this survey paper.

3. MPSoCs in IoT

The MPSoCs have become a de-facto computing platform and they can be used in various computationally extensive real-time applications.

3.1. Multimedia Surveillance

MPSoCs integrated with video and audio sensors are used for target detection and tracking, border protection, public event monitoring (53; 54; 55), video/image enhancement (56), person tracking (57), people and object identification (58; 59). These multimedia surveillance applications involve compression/decompression, encoding/decoding and different conversions techniques which are computationally extensive operations. A few highly complex MPSoCs such as Xetal-I (128 processors) (60) and Xetal-II (320 processors) (61) are also used for surveillance.

Video surveillance to monitor on-road situation is a prime example of the multimedia application in IoT. In this video streaming of on-road situation, the information is collected regarding vehicles positions, traffic jams and road accident severity. In the video streaming the multimedia data content is streamed over the network in an encoded form while the video is displayed to the end user and/or professional either in a recorded or pre-recorded manner. In IoT based applications video streams are usually compressed to reduce the video size and achieve better load balancing in the video communications. The MPEG-encoder is executed numerous times for the whole video stream (62).

3.2. Healthcare and Automated Assistance

Multiprocessor systems are widely adopted by remote medical centres for various advanced healthcare related applications such as patient monitoring (63), drug administration, diagnostics (64; 65), human gait analysis (66), care assistance, and motoring (67). These services using MPSoC are helping to reduce the frequency of patient's visit to the hospitals while enhancing the Quality-of-Life (QoL). STMicroelectronics MPSoC is one of the popular multiprocessor platform that is widely adopted in advanced healthcare (63).

3.3. Environment Monitoring

MPSoCs are also used in applications such as animal and bird tracking as well as condition monitoring for irrigation, livestock, crops, and air pollution (61; 68). Monitoring systems are vital in time-critical applications, such as wild fire containment, flood detection, and disaster management (69; 70). High-performance computing platforms provided by MPSoC produce smart technology to monitor and detect natural and anthropogenic emergencies.

3.4. Industrial Applications

MPSoCs are deployed to extract and analyze information regarding civil structures e.g. nuclear power plants, pipelines, and large bridges especially during and after earthquakes, high winds or environmental changes (71). In the industry, MPSoCs are used for automation and manufacturing process control for example, Xilinx Zynq® UltraScale^{TM} are deployed in robots for operations such as supervision, control, and automation that increase repeatability while reducing human efforts while maintaining continuous operation (72; 73).

Among the applications of MPSoCs in IoT, the most popular is surveillance where video analytics is performed for different purposes. Video analytics also called Video Content Analysis (VCA), it involves different techniques to monitor, extract and analyze the information from video streams (74). Closed-circuit television (CCTV) cameras are the main contributors of computerized video analysis. In video analytics a key challenge is the size of the video data. In one second of high-definition video there is approximately equal to 2000 pages of text. Video analytics in IoT is widely used in the recent years for surveillance and automated security.

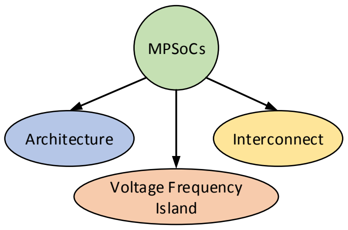

Figure (4): MPSoC systems categorisation

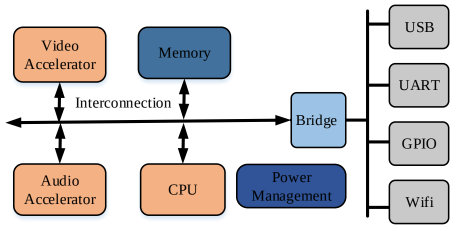

Figure (5): Generic heterogeneous MPSoC architecture

Automated surveillance systems are cost effective, cheaper, and remain focused as compared to labor-based surveillance systems. Video analytics can be used for human recognition, face recognition, object detection, recognizing suspicious activities, and detecting breach of restricted zones (75). In terms of the IoT architecture two known approaches can be adopted for performing video analytics, namely (1) server-based architecture and (2) edge-based architecture. Each architecture is explained as follows: (76).

3.4.1. Server-based Architecture:

In this approach, captured video using cameras is transmitted to the centralized and dedicated server where video analytics is performed. The generated video is usually compressed to reduce the frame rates or the image resolution due to limited bandwidth availability. In this configuration the compressing may result the loss of information which can adversely affect analysis overall accuracy. However, the server-based approach facilitates easier maintenance.

3.4.2. Edge-based Architecture:

In this configuration, analytics are applied at the SN level or 'edge' of the system. In other words video analytics is performed on the raw data gathered from the camera in the SNs. In this approach the entire content/data of the video stream remains available for the video analysis. Therefore, no loss of information occurs and enables efficient and effective content analysis. However, edge-based systems are more costly to maintain and posses lower processing power capability compared to server-based systems.

Briefly, server-based approach is easier to be implemented and maintained while edge-based system is costly though the entire video stream is available for performing video analytic on gathered data.

4. MPSoC Types

MPSoC systems can broadly be categorised into three types based on their architecture, inter-processor communication mechanism and number of processors per voltage island as shown in Figure (4).

4.1. Architecture

In terms of the types of the processors used, MPSoCs can be divided into two general groups of homogeneous and heterogeneous.

4.1.1. Homogeneous MPSoCs

Homogeneous MPSoCs are symmetric multiprocessing systems where identical processing elements with the same Instruction Set Architecture (ISA) are used (77). Therefore, in homogeneous MPSoC, a single thread task will have the same power consumption and complete in the same time irrespective of which processor is utilised. Examples of commercial homogeneous MPSoC include Samsung Mongoose M2 with 4, Cortex-A53 processors and EZchip TILE-Mx100^{TM} with one hundred Cortex-A53 processors. Homogeneous MPSoCs are suitable computing platforms for applications where communication to computation ratio is higher (78).

4.1.2. Heterogeneous MPSoCs

Heterogeneous MPSoC shown in Figure (5) include multiple types of different processing elements. Heterogeneous MPSoCs can either be functional asymmetric or performance asymmetric. Examples of heterogeneous MPSoC include, Samsung Exynos 9810 used in Samsung Galaxy S9+, S9++, and Note 9+. Some other MPSoC and application are listed in Table (3).

1. *Functional Asymmetric:* It contains a set of architecturally different processing units with, as a consequence different ISA. Figure (5) shows a generic functional asymmetric heterogeneous MPSoC consists architecturally different processing units i.e. general-purpose processor (CPU), video accelerator, and audio accelerator (79).
2. *Performance Asymmetric:* The ISA remains the same but performance and power consumption of the processing units are different. Samsung Exynos 5 Octa (big.LITTLE) also known as Exynos 5410 used in Samsung Galaxy S4 is an example of performance asymmetric heterogeneous MPSoC. It has 4 Cortex-A15 processors and 4 Cortex-A9 processor (80).

4.2. Interconnect

The second broad categorisation of MPSoC is interconnect method, the communication infrastructure for inter-processor communication (81). Sub-categories are (1) *Bus-based MPSoC* and (2) *NoC based MPSoC*. Communication network type plays a vital role in achieving energy-efficiency and avoiding communication congestion. Loss of data in the communication network reduces the overall system performance and energy- efficiency (82).

Table (3)
MPSoCs architectures used in smart-phones or IoT

Model	Type	Architecture
Samsung Exynos 9810	Heterogeneous (smart-phone)	4 Mongoose 3 big cores and 4 Cortex-A55 little cores
Apple A11 Bionic	Heterogeneous (smart-phone)	2 ARMv8-A monsoon and 4 ARMv8-A Mistral
Intel® Stratix® 10	Homogeneous (smart-phone)	Quad-core ARM Cortex-A53
Tilera TILE-Gx72™	Homogeneous (IoT)	72 VLIW (Very Long Instruction Word) RISC processors operating upto 1.2 GHz frequency

4.2.1. Bus

The bus-based architecture is probably the oldest on-chip interconnect in the computer industry and is used in many MPSoCs (83). The bus provides a communication mechanism which interconnects different components (processing units, memory, I/O units) of the MPSoC architecture. The bus interconnect shown in Figure (6) is an easier approach to integrating a small number of components due to its simple protocol design and silicon cost. However, it offers limited bandwidth and increased delays when used for a large number of components.

Matrix bus interconnect offers a solution for bandwidth as it offers multiple communication paths. Figure (7) demonstrates an example of 3 master and 5 slave Advanced Microcontroller Bus Architecture (AMBA) bus matrix communication subsystem architecture for dual ARM processor-based MPSoC. A bus matrix (crossbar switch) has several parallel wires (busses) which offer a suitable backbone to support concurrent data streams. The input stage handles interrupted bursts if receiving slaves are unable to accept them immediately. Decode generates a signal for proper slave selection. The component arbiter collects requests from all masters and allows only one module to have access to the slave at a time (84). The evolution of bus interconnect is a progression started from AMBA, Advanced System Bus (ASB) to High-Performance Bus (AHB) then AMBA AHB-Lite and finally AMBA AXI (Advanced Extensible Bus). Among these AXI4 is the latest example of the MPSoC interconnects (85).

Xilinx Zynq Ultrascale+MPSoC, a bus based MPSoC widely used for IoT applications. It deploys ARM AMBA-AXI4 bus to interconnect Quad-core ARM Cortex-A53, Dual-core ARM Cortex-R5 real-time processors, and other controls/peripherals (86; 26). ARM AMBA-AXI4 bus also establishes communication between a memory-mapped master device and a single or multiple memory-mapped slave devices (26). Traditional bus communication architectures support only limited bandwidths and are not scalable for high-performance designs leading to the development of NoC-MPSoC (87; 88).

4.2.2. NoC

It is a network based communication subsystem on a chip. NoC technology applies the method of computer networking and improves the communication mechanism compared to conventional crossbar communication architectures. NoC increases the scalability, flexibility, and power-efficiency of

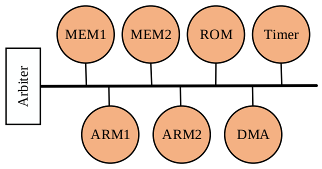

Figure (6): Bus communication architecture

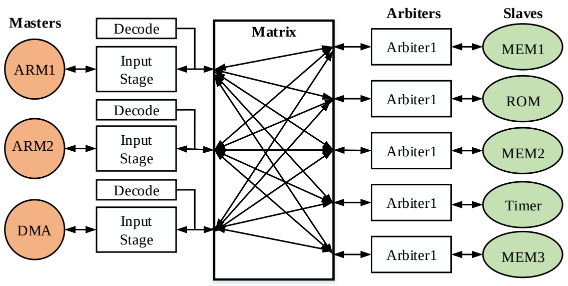

Figure (7): Bus matrix communication architecture

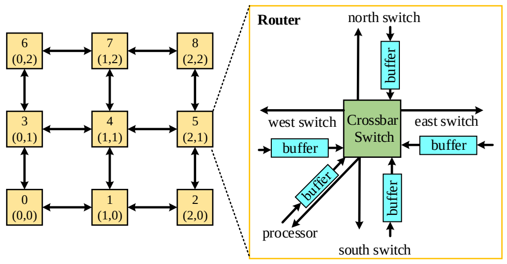

Figure (8): A typical 2D-mesh NoC architecture

MPSoC. Figure (8) shows a typical 2D-mesh NoC that consists $N_R = 3$ rows and $N_C = 3$ columns i.e. a total of 9 routers. Each router in a NoC has five ports associated with buffers, four ports are used to communicate with the neighbour routers and one dedicated for the purpose of communicating with the processing unit. The communication link is used to connect two routers and a router with a processor (11; 89).

In NoC based MPSoC, the various modules such as Processing Elements (PEs) i.e. tiles, IP blocks, and memory elements exchange the data through a network. NoC comprises of point-to-point data links and the interconnection is provided by routers (switches). The NoC based MPSoCs can either be homogeneous or heterogeneous depending upon the

Table (4)
Different DVFS-enabled NoC based MPSoCs from Tilera™

MPSoC Model	Network Topology	Max: CPU Clock Rate
Tile64 ™	Mesh	600-900 MHz
TilePro64 ™	Mesh	600-866 MHz
TilePro36 ™	iMesh	500 MHz
Tile-Gx ™	iMesh	1.2 GHz
Tile-Gx36 ™	iMesh	1.2 GHz
Tile-Gx16 ™	iMesh	1.2 GHz
Tile-Gx ™	iMesh	1-1.2 GHz

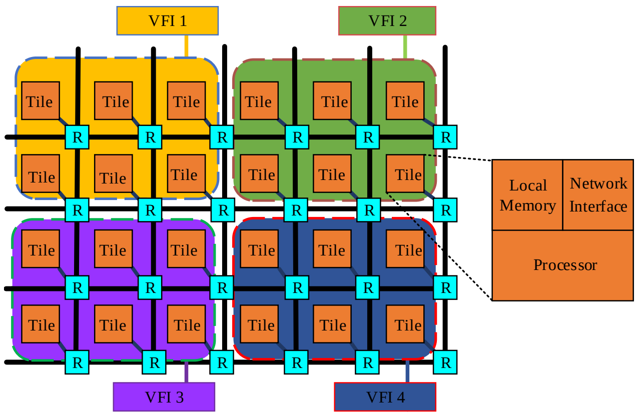

Figure (9): A generic representation of VFI based NoC-MPSoC

nature of the PEs used (90). Samsung Exynos 7 Octa (7420) with 2.1GHz Quad-Core Cortex®-A57 and 1.5GHz Quad-Core Cortex®-A53, HiSilicon Kirin 960 with 4× Cortex-A73, 4× Cortex-A53 are examples of NoC based MPSoC. Various other NoC based MPSoCs by the Tilera corporation are listed in Table (4) (91; 92).

4.2.3. *Voltage Frequency Islands based MPSoCs*

More recently, Voltage Frequency Island (VFI), Globally Asynchronous Locally Synchronous (GALS) model is introduced to NoC interconnect, where the tiles are partitioned into islands while each island is optimised with its own threshold voltage, operating frequency, and supply voltage. MPSoC systems implemented with GALS have a reduced number of voltage level converter and mixed-clock/mixed-voltage FIFO requirements (93; 94; 89). Figure (9) shows a generalized VFI based NoC-MPSoC. It consists of four VFIs represented by different colours (yellow, green, violet, and blue) and six tiles per VFI. Each VFI contains an independent voltage supply and a local clock. Inter-VFI communication is established through mixed-voltage FIFO, mixed frequency clocks, and voltage converters. Moreover, each tile in every VFI has a local memory, network interface, and a processor (95). It is worthy of note that state-of-the-art commercially available multiprocessor systems e.g. Intel Itanium i7 and IBM Power 7 series use VFI based MPSoC architectures (96; 97).

5. Models and Scheduling

This section presents the relevant application models used in the literature for energy-aware scheduling algorithms deploying MPSoCs. Specifically, Section 5.1 overviews various application models, Section 5.2 presents a power model and Section 5.3 discusses task scheduling and its different types.

5.1. Application Model

Workload/application (98; 99) in the literature is represented either by an independent or dependent task models. There are no precedence constraints in the independent task model.

5.1.1. *Independent Task Model*

Independent task model $T = \{T_1, T_2, T_3 \dots T_n\}$ is a collection of tasks with no inter-task data dependencies i.e tasks are not related by precedence relations. In the task model, n shows the total number of tasks. Each task T_i has an execution time, t. Within the independent task model each task T_i contains all the necessary data required to execute on a processor (100; 101; 102). Independent tasks can be executed on the processors in any order by the scheduler. However, a technique called job scheduling is often used to allocate the tasks on available distributed processors such that the overall makespan is reduced. Specifically, scheduler priorities some tasks over others to minimise the finishing time of the last task T_n (103).

Independent task models are used for safety-critical applications e.g. autonomous vehicles. Moreover, different applications running concurrently such as video streaming, target tracking, and image enhancement can be modelled as independent tasks (104).

5.1.2. *Dependent Task Model*

The dependent or interacting task model is mostly represented by a Directed Acyclic Graph (DAG) shown in Figure (10). This popular representation of an application comprises characteristics such as inter-task communication data size, tasks deadlines, and task dependencies. A DAG can be mathematically represented as $G(V, E, \tau)$ where, $V = \{v_1, v_2, v_3 \dots, v_n\}$ shows the tasks in an application/workload, $E \subseteq V \times V$ denotes data dependencies between the tasks. τ shows edge weights. The edge weight is basically the data transferred (represented by the numbers on each edge) in units of bits between two nodes v_i and v_j (105; 106; 107). Conditional Task Graph (CTG) is another-type-of dependent task model. In the CTG, an edge is called a conditional edge if it is associated with a condition representing that the following task is executed only if the condition holds. An edge is an unconditional edge if no condition is associated with it (108).

5.2. Power Model

MPSoCs are CMOS devices, the overall power consumption in CMOS circuits is due to dynamic and static power

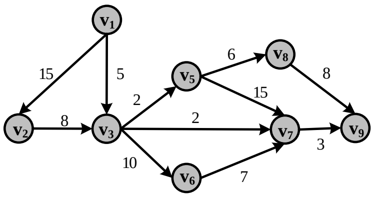

Figure (10): Directed acyclic graph

dissipation. Dynamic power dissipation occurs due to transistors switching and dominates the total power consumption in CMOS technology. The dynamic power consumption of each processor in MPSoC executing a task at a certain discrete voltage and frequency level (V_{dd}, f) is given as follows (109; 11):

$$P_D = C_{eff} \times V_{dd}^2 \times f \tag{1}$$

where V_{dd} represents the supply voltage and f denotes the operating frequency while C_{eff} is the effective switching capacitance. The cycle length t_{cycle} of the clock for executing a task assigned to a processor using a certain speed/voltage level can be represented as $t_{cycle} = \frac{L_d \times K_6 \times V_{dd}}{(V_{dd}-V_{th})^{\alpha}}$. Where K_6 denotes technology dependent constant, L_d shows the average logic depth of the processor's critical path, while 1.4 $\leq \alpha \leq$ 2, and V_{th} is the threshold voltage which can be calculated as $V_{th} = (V_{th_1} - K_1) \times (V_{dd} - K_2) \times V_{bs}$, where V_{th_1}, K_1, K_2 are technology dependent constants and V_{bs} represents body bias voltage.

Now, suppose I_{subn} denotes subthreshold leakage current while I_j shows the junction current, and L_g represents the total number of devices connected in the circuit, then the static power (P_s) can be expressed as follows:

$$P_S = L_g \times (V_{dd} \times I_{subn} + |V_{bs}| \times I_j) \tag{2}$$

where $I_{subn} = K_3 \times e^{K_4 V_{dd}} \times e^{K_5 V_{bs}}$ and K_3, K_4 and K_5 represent processor technology specific parameters (constants). The total power consumption of each processor can be computed as follows:

$$P = P_D + P_S + P_{on} + P_{sleep} + P_{soh} \tag{3}$$

where P_{on} is the idle power when the processor is not executing any workload i.e. P_{idle}. The P_{soh} indicates sleep overhead power to switch a processor into sleep mode and vice versa. The break-even time T_{be} to put a processor into a low-power mode when possible can be calculated as $T_{be} = max\left(\sigma, \frac{E_x - \sigma . P_x}{P_{ref} - P_x}\right)$. Where σ and E_x show the initial and final energy transition overheads respectively. P_{ref} denotes the processor's power consumption in a default state when no task executes while P_x represents low-power state.

5.3. Task Scheduling

Task mapping and scheduling is a process of properly allocating a set of tasks on the processors such that specific obligations are fulfilled e.g. power consumption optimisation and/or execution time reduction. Proper task mapping and scheduling approach can drastically influence an embedded system's performance and reliability (110). Task mapping and scheduling on MPSoCs is NP-hard problem (105; 111). Therefore, various scheduling algorithms are proposed by the researchers using Non-Linear Programming (NLP) (112), Mixed Integer Linear Programming (MILP) (109), and Integer Linear Programming (ILP) (113). Numerous other search-based approaches are also investigated using Differential Evolution (DE) (114), randomisation (115), Simulated Annealing (SA) (10), Particle Swarm Optimisation (PSO) (116), and Genetic Algorithm (GA) (117; 11). These algorithms are categorised in Figure (11).

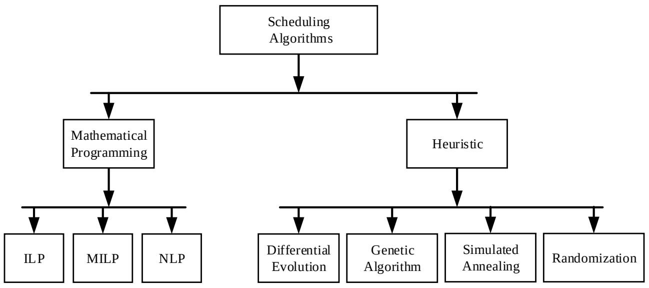

Figure (11): Task scheduling algorithms

5.3.1. *Scheduling and Mapping Types*

Dependent task scheduling and mapping can be divided into two other types dynamic and static given as follows:

1. *Static Scheduler* assigns task priorities before the embedded system runs. Static scheduling simplifies optimisation complexity, although this technique is inefficient regarding resources utilisation (118). It is easy to be implemented and guarantees to meet the tasks deadline (119).
2. *Dynamic Scheduler* executes the tasks on the processing elements in real-time. The dynamic scheduler considers the available resources and can rearrange the tasks list during runtime (118; 120). The disadvantage of dynamic scheduling is the runtime overhead and there is no guarantee that all the tasks to be executed can meet their deadline (119).

6. System Level Power Management Techniques

In this section, we present an overview of the system level power management techniques shown in Figure (12).

6.1. Dynamic Voltage Frequency Scaling

DVFS is a system level technique whereby the processor supply voltage and clock frequency are dynamically reduced as a means to reduce overall power consumption without negatively impacting performance and deadline completion (121; 122). Modern MPSoCs includes DVFS-enabled

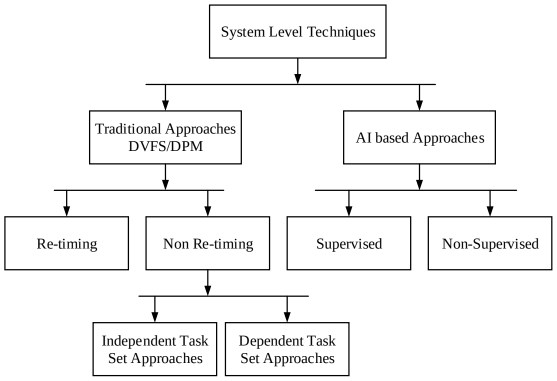
System Level Techniques
Traditional Approaches DVFS/DPM
AI based Approaches
Re-timing
Non Re-timing
Supervised
Non-Supervised
Independent Task Set Approaches
Dependent Task Set Approaches

Figure (12): System level power management techniques categorisation

Table (5)
The 70 nm processor technology parameter values

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 (6)
Transmeta crusoe processor power consumption at different discrete supply voltage levels

Parameters	Value				
V_{dd}(V)	0.85	0.80	0.75	0.70	0.65
f(GHz)	2.10	1.81	1.53	1.26	1.01
P_D(mW)	655.5	498.9	370.4	266.7	184.9
P_S(mW)	462.7	397.6	340.3	290.1	246.0

processors. As an example, the Samsung Mongoose M2 has 4 DVFS-enabled homogeneous processors with an operating frequency range of 2.3 to 2.8 GHz. Reducing the supply voltage can significantly cut the energy consumption in MPSoCs because voltage has a quadratic law relationship with energy consumption (123). Herbert and Marculescu experimentally analysed the impact of DVFS on energy consumption for MPSoCs (124).

In order to understand the working principle of DVFS and its impact on power-efficiency we consider 70 nanometer (nm) technology parameters listed in Table (5) (109). According to the specification, a Transmeta Crusoe processor (125; 126) which operates at five discrete voltage levels of $\{0.65, 0.7, 0.75, 0.8, 0.85\}$. The value for body bias voltage is $V_{bs} = -0.70$ V. Given this data we calculate the corresponding frequency, dynamic power P_D and static power P_S using equations given in Section 4.2 for the different supply voltage V_{dd} levels summarised in Table (6). Both P_D and P_S considerably reduce with the decrease in the V_{dd}.

Table (7)
Two different processors power consumption modes

Transmeta Crusoe		PXA-250	
Parameter	Value	Parameter	Value
P_{idle}	276 mW	P_{idle}	555 mW
P_{sleep}	80.0 μ W	P_{sleep}	180 μW
P_{soh}	385 μ W	P_{soh}	483 μ w

6.2. Dynamic Power Management

DPM is another energy-saving technique that can be emmployed without significant loss of performance. The objective of DPM is to switch the system to low-power mode when idle, returning to full power mode when required (127). In other words, the DPM technique switches the processor to an inactive state (low-power) for as long as possible, while ensuring that all viable tasks finish within their deadlines. In CMOS technology the power consumption is due to both dynamic (electronic switching) and static (electronic leakage) components. The objective of DPM is to reduce static power consumption while DVFS is used to minimise dynamic power consumption (27).

Table (7) shows different power consumption modes of Transmeta Crusoe and PXA-250 processors. Power consumption in sleep (low-power) mode is evidently smaller than idle mode, therefore, a significant amount of energy can be saved using DPM technique to switch an idle processor into a low-power mode whenever possible.

6.3. Coarse-grained Software Pipelining/Re-timing

Many applications such as streaming execute repeatedly. Such applications and are known as periodic applications. Periodic applications are associated with an integer value called the period that defines the time interval after which the application executes again.

Consider a periodic application modelled by a DAG as shown in Figure (13)(a). Figure (13)(b) shows the schedule for the first three periods. Notice that v_2 cannot execute until v_1 completes execution because v_2 is dependent on v_1. Thus the processor where v_2 is scheduled remains idle (assuming the idle interval is shorter than break even time) during the time interval v_1 executes. In other words, the available slack (on the processor where v_2 is scheduled) is wasted. The wasted slack can be minimised by a technique called re-timing.

Re-timing is a technique that minimises the wasted slack by regrouping nodes from different periods (128). Consequently, the intra-period precedence constraints are transformed into inter-period precedence constraints. Figure (13)(c) shows the schedule where the execution of v_1 and v_2 is delayed by one period. Since v_1 executes one period ahead of v_2, it can start executing early as shown in Figure (13)(c). However, re-timing has a side-effect i.e. adds a prologue.

Table (8)
Independent tasks system level energy-aware approaches

Reference	Platform	Classification	Pros	Cons
(129)	Homogeneous Multiprocessors	Mapping & scheduling	Minimises the processing energy and schedules the tasks using EDF policy	Simplified power model and does not consider the static power and integration of DVFS/DPM. It merely generates tasks list schedule
(130)	Homogeneous bus based MPSoC	Online scheduling	Reduces the processing energy and utilizes the run-time slack	Considers ideal platform and simplified power model with negligible static power consumption
(116; 117)	Heterogeneous bus based MPSoC	Offline scheduling	Extensive solution space searching for processing energy reduction	Considers dynamic energy only and assumes negligible static energy and does not use DVFS only considers task allocation
(131)	Homogeneous bus based MPSoC	Online mapping	Feedback control scheme for proper task allocation to reduce processing energy reduction	Does not utilize the processors full capacity and some of the workloads execute before their deadlines
(132; 133)	Homogeneous VFI based MPSoC	Offline mapping	Dynamic and static power consumption reduction and efficient algorithms with lower complexity	Assumes an ideal MPSoC platform with negligible consideration of transition overhead
(134)	Heterogeneous VFI based MPSoC	Online mapping	Reducing the processing energy while maintaining the required performance through appropriate task mapping	The increased transition overhead and other drawback is slow response time

7. System Level Energy Optimisation Approaches/Algorithms

In this section, we review mapping and scheduling approaches/algorithms on multiprocessor systems for energy management. These approaches are applicable to both SNs in IoT and consumer electronics.

7.1. Independent Task Set

Energy optimisation approaches using MPSoCs platform for independent task models are listed in Table (8) with their advantages and drawbacks.

Aydin et al. (129) developed an algorithm with $O(n^2 log n)$ complexity. It is one of the first DVFS based energy-efficient scheduling algorithms for independent (periodic) real-time

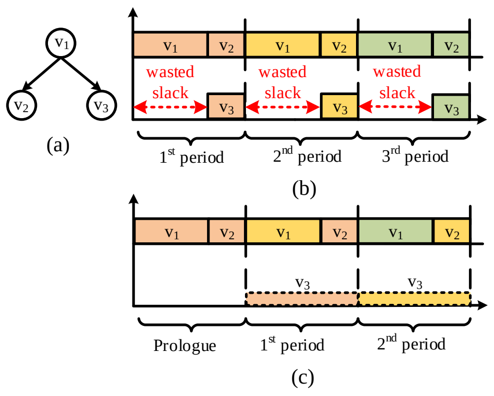

Figure (13): Coarse-grained software pipelining (a) a simple DAG with three task nodes representing a workload (b) tasks mapping without re-timing (c) task mapping with re-timing

tasks with different power consumption characteristics on multiprocessor systems. The authors formulated the scheduling problem as an NLP problem and assigned constant speed (voltage and/or frequency) to the tasks while maintaining the optimality. They also showed that Earlier Deadline First (EDF) based task scheduling can generate a feasible schedule while $O(n^2 logn)$ would determine the optimal speed values for the independent tasks.

Zhu et al. (130) proposed a DVFS based online speed assignment algorithm known as Global Scheduling with Shared Slack Reclamation (GSSR) for the independent frame based task set under implicit deadlines. The GSSR algorithm is called each time when a task completes or a new frame starts. It determines the minimum speed required to run the task on MPSoC architecture without violating the frame deadline. The simulation results for automated target recognition (ATR) and Berkeley MPEG-1 encoding produced energy savings of up to 44% when compared to system level approach called static power management (SPM) proposed in (135) by Gruian.

Zhang et al. (116) developed a meta-heuristic based Shuffled Frog Leaping Algorithm (SFLA) inspired by the evolution of frog foraging for convergence acceleration and overall energy consumption reduction. SFLA integrates the benefits of PSO and Memetic Algorithm (MA) to mix information from local searches and to move towards a global solution. They considered independent periodic tasks and heterogeneous MPSoC system and based on SFLA designed a scheduling algorithm to meet hard task deadlines and reduce the overall energy consumption. Their approach outperformed ACO, GA and achieved 30% and 40% energy-efficiency.

Kumar and Vidyarth (117) integrated task mapping and voltage assignment in a single optimisation loop of GA. Their approach searches the solution space for near optimal task mapping. They used the DVFS technique to assign voltages to the tasks such that the dynamic energy consumption is minimised with acceptable performance trade-off. The energy savings further increased when makespan is extended. The authors reduced the energy consumption by 54.6% to 59.4% compared to other non-DVFS HEFT (136) and Genetic Algorithm-Struggle (GA-ST) (137) approaches.

Dziurzanski and Singh (131) suggested a DVFS based feedback control scheme for task allocation on MPSoC system to achieve higher energy-efficiency. The developed admission control algorithm performs a schedulability analysis considering the previous platform states and rejects the tasks being expected to violate their deadlines.

Energy optimisation approaches deploying VFI based NoC-MPSoC for task mapping and scheduling using a set of independent tasks are proposed by the authors in (133; 132; 134). Pagani et al. (133) developed a scheme called Single Frequency Approximation (SFA) for optimal frequency and voltage assignment to each island in the MPSoC system. They developed a dynamic programming mapping algorithm integrated with SFA and reduced the energy consumption and running time. Liu and Guo (132) presented a Voltage Island Largest Capacity First (VILCF) algorithm for mapping and scheduling the tasks in order to increase energy-efficiency. The algorithm aimed to first fully use an active island before activating more islands in the MPSoC platform. Singh et al. (134) proposed an energy-efficient run-time management approach for task mapping on heterogeneous VFI based NoC-MPSoC and threads partitioning of the concurrent applications. The authors integrated GPU into the computing system and performed proper workload distribution between GPU and CPU using profiling knowledge.

7.2. Dependent Task Set

Other researchers have investigated various scheduling and mapping approaches using dependent task model to optimise energy consumption. In this section we discuss some well known MPSoCs energy management approaches in details as summarised in Table (9). The energy consumed (both static and dynamic) by the processors and communication network is referred to as processing and communication energy respectively. We categorise the approaches designed for energy-aware scheduling of dependent task set on MPSoCs into (1) communication energy (2) processing energy (3) total energy.

7.2.1. Communications Energy

Carvalho et al. (81) scheduled the dynamic workload on heterogeneous NoC based MPSoC architecture using Nearest Neighbour Heuristic (NNH) algorithm. The authors mainly focused on improving the NoC energy consumption reduction by reducing average link occupation and minimised the links congestion using Path Load (PL) algorithm. Wang et al. (113) optimised the communication energy for streaming applications deploying a bus based MPSoC. They reduced the scheduling length and minimised the inter-core communication overhead. The authors formulated task scheduling on the processors as an ILP problem and reduced the overall makespan by minimising the communication overhead and utilized idle slacks in the MPSoC. Similar work is performed by Wang et al. (143) for streaming application. Maqsood et al. (138) mapped the tasks on NoC based MPSoCs

Table (9)
Dependent tasks system level energy-efficient approaches

Reference	Platform	Classification	Pros	Cons
(81)	Heterogeneous NoC based MPSoC	Dynamic mapping	Minimises the communication energy and avoids congestion at the NoC links	The system is not scalable and does not consider the communication weights between parent and child nodes.
(113)	Homogeneous bus based MPSoC	Static scheduling	Optimises inter-core communication energy and reduces the schedule length to improve the memory usage.	Transition overhead and sleep overhead is increased
(138)	Homogeneous NoC based MPSoC	Dynamic mapping	Reduces the communication energy and end-to-end latency	NoC links are not scalable and consumes higher energy during the communication process and task migration overhead exists in the mapping
(139; 140; 141)	Homogeneous NoC based MPSoC	Dynamic mapping	Optimises the inter-core communication for energy consumption reduction and improves the system's fault tolerance.	High processing energy consumption due meeting all the tasks deadlines and high links energy consumption because are not scalable. Moreover, the platform is application specific
(109)	Homogeneous MPSoC	Static scheduling	Integrates DVFS and DPM to reduce processing energy consumption	Does not consider resource constraints and communication energy overhead
(107)	Heterogeneous MPSoCs	Static scheduling	Thermal-aware processing energy optimisation	Slow response time and high communication energy consumption
(142; 108)	Heterogeneous NoC based MPSoC	Static mapping	Processing energy optimisation and NoC links congestion improvement	Does not consider online task scheduling and mapping
(10)	Heterogeneous NoC based MPSoC	Static mapping	Processing energy and communication energy consumption improvement	Does not consider leakage/static power and large execution time
(112)	Homogeneous MPSoC	Online+Offline scheduling	Reduces the total energy consumption	Considers and ideal platform and assigns continuous voltages for the tasks executions. Moreover, transition overhead exists in the online scheduling

using Communication-aware Packing based Nearest Neighbour (CPNN) algorithm. The reduction in communication energy was attained by migrating the tasks from the processors with a light load to other appropriate processors that can actively accommodate those tasks as a consequence the inter-processor communication workload is reduced. The work based on CPNN is further improved by Chatterjee et al. (139) where the authors performed communication energy-aware dynamic task scheduling on homogeneous NoC based MPSoC system for real-time applications. They developed an algorithm Deadline and Energy Aware Mapping and Scheduling (DEAMS) that allocated the resources intelligently to improve the deadline satisfaction rate for the applications with minimum energy consumption. Using the same model in another paper, Chatterjee et al. (140) exploited the slack times in the scheduling mechanism to meet the tasks deadline constraints and optimise the energy consumption while exploiting the idle slack in the processor for fault-tolerant task mapping.

Although these approaches may optimise energy consumption but are limited in only reducing the communication energy and, therefore are only suitable for applications with intensive communication volumes.

7.2.2. Processing Energy

Many energy-aware approaches have been proposed for scheduling set of tasks with precedence constraints with an objective to optimise the processing energy consumption.

Shin and Kim (144) performed energy-aware task scheduling on DVFS-enabled multiprocessor systems. The authors designed a condition-unaware task scheduling algorithm integrating the task ordering algorithm for CTGs to reduce the computational energy consumption. This algorithm assigned start time and the clock speed to each task considering task execution profiles and condition matching. An NLP based voltage assignment is deployed to assign discrete voltages to each task. Chen et al. (109) used DVFS and DPM techniques for real-time task scheduling on MPSoC platform and formulated the processing energy optimisation problem using MILP. The idle intervals of each individual processor were utilised and optimal non-preemptive, time-triggered tasks schedule was generated. Tariq et al. (112) proposed a two-phase offline task scheduler targeting homogeneous MPSoCs system with shared memory to minimise the total worst-case utilisation of the processors. The authors assigned optimal task execution speed using convex NLP. Moreover, they also developed an online task speed assignment mechanism using a Dynamic Voltage Scaling (DVS) heuristic algorithm for processing energy consumption reduction. Zhou et al. (107) developed a two-level task scheduling approach deploying DVFS-enabled heterogeneous MPSoC for energy consumption reduction under task precedence constraints. In step one, the MPSoC model is transformed into a virtual multiprocessor model that supports a single fixed frequency only. Secondly, the tasks are assigned to the processors of the MPSoC system considering the precedence constraints of the tasks.

The approaches discussed so far either aim to minimise the processing energy or communication energy only. Next, we briefly survey approaches that minimise both processing and communication energy.

7.2.3. Total Energy

Gosh et al. (142) presented a unified approach for task mapping problem on heterogeneous NoC based MPSoC with near optimal solution time and heuristic solution. Mapping is performed using MILP and tasks are mapped such that links congestion does not occur and tasks are executed on optimal voltage/speed levels. Huang et al. (10) used an extended ILP formulation for optimising both the communication and processing energy on heterogeneous NoC based MPSoC architectures. Moreover, task scheduling is accelerated using Simulated Annealing with Timing Adjustment (SA-TA) heuristic algorithm. The SA-TA algorithm basically optimises the energy consumption by reaching near to the global optimum under even tight timing constraints. Abdel-Basset et al. (145) and Deng et al. (146) have proposed a modified the whale optimisation algorithm with two objectives minimizing the total energy consumption and the makespan.

In all these approaches it is assumed that only processors are voltage scalable. Therefore, the DVFS approaches allocate all the slack to tasks only and the communication energy is only reduced through task mapping. Andrea et al. (147) and (148) have shown that if like processors, communication links are voltage scalable, more energy can be saved by sharing the available slack between communication and task.

Andrea et al. (147) and (148) propose an NLP and a MILP based DVFS algorithms for a tasks set with precedence constraints on heterogeneous MPSoC. Their proposed approach shares available slack between task and communication nods such that total energy consumption is minimized. Shin et al. (149) consider a NoC based MPSoC model with voltage scalable links and assume that processors operate at fixed frequency and voltage levels. They propose energy efficient voltage scaling algorithm that aims to minimize the communication energy by statically assigning voltages and frequencies to links. Li et al. (150) propose task mapping, scheduling and DVFS algorithm for a task set with precedence constraints on homogeneous NoC based-MPSoC model with voltage scalable links and processors. They propose a two-step approach. In the first step, they propose a quadratic programming based mapping algorithm that maps tasks to a processor such that total weighted communication distance is minimized. In the second step, they use GA to assign voltages and frequencies to tasks and communications. Tariq et al. (108) investigated the task scheduling and mapping on NoC based MPSoCs for total energy consumption reduction. They developed a heuristic algorithm to construct a single unified schedule for a set of tasks and assigned integer frequencies to both the tasks and communication using ILP. ILP collectively optimised the voltage of the processors and NoC links. Ali et al. (11) developed a contention-aware integrated task mapping and voltage assignment (CITM-VA) static energy management scheme and minimised the processing energy consumption while explicitly considering the contention at the NoC links.

Here we explain how GALS is better and then discuss few approaches of GALS. Recently GALS paradigm is introduced to NoC interconnect to group the tiles into islands. Each island is optimised with its own operating frequency, threshold voltage, and supply voltage to reduce power overhead. VFI based multiprocessor architectures are suitable for data-extensive applications due to their higher energy-efficiency and performance, and lower complexity. VFI based NoC-MPSoC curtails the overall computing architecture complexity by decreasing the number of multiple clock first-in-first-out (MCFIFO) and voltage level converters (VLCs) (151). Ninomiya et al. developed a task scheduler for VFI based NoC-MPSoC architecture using simulated annealing algorithm. They generated task schedule considering the application deadline such that the overall energy consumption is reduced while network traffic and congestion are also decreased (152). Han et al. investigated a static mapping and scheduling scheme on VFI based NoC-MPSoC architecture for tasks with precedence constraints in order to minimise the makespan and inter VFI communication for total energy consumption (89). Tariq et al. developed a scheduling algorithm referred to ARSH-FATI to perform task mapping, ordering, and voltage assignment in an integrated manner. The ARSH-FATI static scheduler also considered processors energy performance profiles and available discrete voltage/frequency levels within processors and inter-VFI communications (151). Tariq et al. in another investigation developed a meta-heuristic that has the capability to switch dynamically between exploitation and exploration search modes at run-time for better energy trade-off. They integrated communication contention-aware Earliest Edge Consistent Deadline First (EECDF) scheduling within the meta-heuristic to achieve higher total energy consumption reduction (153).

7.3. Task level Coarse-grained Software Pipelining or Re-timing

In this section, we review energy-aware re-timing based scheduling techniques integrated with DVFS deploying MPSoC architectures as listed in Table (10).

Re-timing is extremely effective in optimising energy consumption when integrated with other system-level approaches such as DVFS and DPM. It works by re-scheduling a parent node a few periods before the child node. Consequently, the parent task executes a few periods ahead of the child node and the data needed by the child node from the parent node is stored in a buffer. When the child node executes in the period it is scheduled to execute it can access the stored data without having to wait for the parent node to complete in that period. In short, re-timing transform the dependent task set into an independent task set. Integrating re-timing with DVFS, DPM or both can significantly reduce energy consumption because there are no Intra-period data dependencies and the slack that is otherwise wasted due to these dependencies can be utilized for energy optimisation.

Kim et al. (158) proposed a re-timing based energy optimisation technique to minimise the energy consumption for uniprocessor systems. Their approach is designed specifically for uniprocessor systems and is not applicable to multiprocessor systems. Shao et al. (156) integrated DVFS technique and pipelining with loop scheduling approach using an MPSoC platform in order to optimise the energy consumption. This Loop optimisation is based on instruction-level pipelining thereby, it is not applicable to periodic dependent task set. Unlike the techniques implemented by (156) that integrated DVFS with coarse-grained software pipelining, Wang et al. (128) and (154) propose approaches to schedule dependent task set on multi-processor systems and optimise the energy consumption by combining task-level coarse-grained software pipelining with DVFS. Wang et al. (128) utilize software pipelining to optimally remove interprocessor communication overhead. They propose Mixed Integer Linear Programming-based (MILP) called Joint Computation and Communication Task Scheduling (JCCTS) algorithms to regroup nodes from different periods. The objective of JCCTS is to optimally remove communication overheads such that the latency overhead due to re-timing is minimized. They have shown that JCCTS can significantly improve energy consumption when combined with a DVFS. Liu et al. (155) propose an algorithm called RDAG and Wang et al. (154) used the same algorithm to transform the dependent task set into an independent task set. Wang et al. (154) have proposed an algorithm called GeneS that solves the problem of task mapping, ordering and voltage assignment in an integrated manner whereas Liu et al. (155) developed an algorithm known as Springs that reduced the schedule length by assigning higher voltages provided that the retiming constraint is smaller than the schedule length and vice versa. Although RDAG is an effective re-timing timing technique but it has been shown by Tariq et al. (153) and (157) that it unnecessarily increases prologue latency or re-timing delay. The primary objective of these approaches is to minimise the prologue latency and neglect memory overhead caused by re-timing.

Streaming applications executing on MPSoCs may require large buffers to store intermediate processing results. As a result these buffer arrays account for a significant portion of application binary footprint (143). The memory consumption further increases due to re-timing because of the buffers needed to store the data needed across different periods. Wang et al. (143) proposed a MILP-based algorithm called Memory-Aware Optimal Task Scheduling (MAOTS) and a heuristic algorithm called Heuristic Memory-Aware Task Scheduling (HMATS). The objective of both algorithms is to regroup tasks and communications such that the interprocessor communication overhead is reduced and the memory overhead is minimised. Although both MAOTS and HMATS try to reduce the memory footprint of re-timing but they are designed to optimise the schedule makespan rather than the energy consumption optimisation.

Tariq et al. (153) proposed energy and memory aware software pipelining approach. Their approach is different than the existing approaches in that energy is optimised along with ensuring to satisfy the memory capacity constraints. They developed a scheduling algorithm called memory-aware re-timing conditional task graph (EMRCTG) that integrated re-timing with DVFS for CTGs. EMRCTG works in two phases. First, it transforms intra-period data dependencies into inter-period dependencies and generates task schedule using NLP while considering an infinite memory capacity. Then the authors analyse the memory consumption for the generated schedule and initiate schedule repair if violation in the memory capacity constraints occurs. The schedule repair determines a set of nodes to reduce their re-timing values such that the memory capacity constraint satisfies.

Table (10)
Coarse-grained software pipelining integrated with DVFS and/or DPM for energy consumption reduction

Reference	Platform	Mechanism	Pros	Cons
(154)	Homogeneous bus based MPSoC	Eliminates the idle slacks in the schedule occurred due to precedence constraints between the tasks and inter-processor communication overhead	Energy, memory aware multiprocessing platform for data intensive applications and overall throughput improvement.	Increased prologue
(128; 155)	Homogeneous bus based MPSoC	Reducing communication overhead from the schedule length and optimising the overall schedule makespan for streaming applications by transforming the intra-period dependencies in the data into inter-period dependencies	Energy optimisation and latency overhead minimisation	Additional memory overhead
(143; 156)	Homogeneous bus based MPSoC	Reducing the schedule length that is makespan effectively	Energy consumption and communication overhead reduction	Longer prologue
(153)	Homogeneous NoC based MPSoC	Combining re-timing with DVFS for CTGs to reduce the total energy consumption	Avoiding memory constraints violation	Memory overhead
(157)	Heterogeneous NoC based VFI-MPSoC	Integrating re-timing with DVFS for CTGs to reduce the total energy consumption	Reducing prologue for streaming applications	Memory constraints violation

Tariq et al. (157) developed an algorithm called ALI-EBAD for CTG to schedule tasks on GALS. ALI-EBAD performed task scheduling and discrete voltage assignment in an integrated manner to achieve higher energy-efficiency. They also reduced latency by using a novel re-timing approach referred to as R-CTG. The R-CTG efficiently minimised the latency without an increase in energy-efficiency.

7.4. AI based Scheduling

In this section we discuss energy-aware task schedulers developed using Artificial Intelligence (AI) for multiprocessor architectures.

Bhatti et al. energy-aware scheduling is one of the earliest approach that deployed Machine Learning (ML). The authors designed an energy-aware scheduling ML based al-

Table (11)
Summary of different energy optimisation approaches/algorithms

Reference	Communication Energy Optimisation	Processing Energy Optimisation	Re-timing + DVFS/DPM	Scalable Link and/ or Contention	Task Model (Application)
(129; 130; 131; 116; 133; 132; 134; 117)	×	✓	×	×	Independent
(81)	✓	×	×	✓	Dependent
(156)	✓	✓	✓	×	Dependent
(142)	×	✓	✓	✓	Dependent
(155)	×	✓	✓	×	Dependent
(113; 140; 139; 138)	✓	×	×	×	Dependent
(154; 128; 143)	✓	✓	✓	×	Dependent
(141; 10; 109; 112)	×	✓	✓	×	Dependent
(108)	✓	✓	✓	✓	Dependent
(107)	×	✓	✓	×	Dependent

gorithm referred to as Hybrid Power Management (HyPowMan) for periodic real-time tasks considering multiprocessor platforms with DPM and DVFS capabilities. HyPowman adopts existing policies for a given set of conditions, and applies them at run-time on other workloads such that energy consumption is reduced. The drawback of this approach is that HYPowMan evaluates the performance of every input which can lead to a high computational overhead (159).Juan et al. designed a semi-supervised reinforcement learning based task scheduler performing dynamic DVFS to reduce the overall power consumption of the multiprocessor system and increase its performance (160). Shen et al. presented a dynamic power management technique (online) based on model-free constrained reinforcement learning. This learning algorithm runs without requiring prior information regarding the the system model and/or workload while dynamically adapts to the environment for achieving autonomous dynamic power management. However, this approach does not consider the variation of application performances for example, page loading rate for browsers and frame rate for video decoders etc. (161). Khan et al. used cooperative RL for energy-aware online task scheduling and determined the next task to be executed using observed application behavior. This RL based scheduling algorithm learns the best possible task scheduling strategy such that a better trade-off between energy and performance is achieved (162). Pham et al. developed a three staged (scheduling, placement and post-placement) supervised learning task scheduler. GA and linear regression are integrated to build a framework for optimising the leakage power. First, a list-scheduling algorithm with priority function is deployed then a cost function at the placement stage is used to manage trade-off between leakage waste and performance. Finally, at post placement stage a heuristic is deployed to further improve the leakage power by closing gaps between reconfiguration and execution of tasks (163). Dambreville et al.(164) developed an online task scheduling algorithm for servers that consist of multi-processors to increase the energy-efficiency of the computing cloud architecture. The reduction in overall energy consumption of the computing platform is achieved by predicting the workload and the jobs are scheduled on the available servers deploying Predict Optimise Dispatch (POD) algorithm. Khan and Xia developed energy-efficient task scheduler using online reinforcement learning to achieve higher performance and energy consumption trade-off. The authors compared their performance in terms of energy with reinforcement learning (RL), distributed independent reinforcement learning (DIRL), exponential weight for exploration and exploitation (Exp3) and cooperative reinforcement learning (CRL) (165). Makrani et al. developed a proactive online resource provisioning approach in heterogeneous cloud platforms for IMC workloads. First, time series neural network is used for predicting the next phase of application then Our artificial neural networks estimates power consumption and performance of the predicted phase of application on different server configurations (166). Esmaili and Pedram developed Deep-EAS scheduler using deep reinforcement learning that performs energy-aware scheduling for workloads having different characteristics without initially requiring to know anything regarding the scheduling task to be executed (167). Yu et al. developed an algorithm referred to as Power Measurement Utility for a Reinforcement Learning (PMU-RL) for workload scheduling on heterogeneous computing platforms to reduce the overall energy consumption (168). Qin et al. designed Energy-aware Multi-objective Reinforcement Learning (EnMORL) scheduling algorithm to reduce for cloud environment. RL based on the Chebyshev scalarisation function is deployed to solve weight selection problem for ensuring the feasibility of the scheduling solutions. The EnMORL is compared in terms of energy-efficiency with two multi-objective meta-heuristics considering four different workflows (169).

8. Challenges and Future Trends

Table (11) presents a summary of energy-efficient task scheduling approaches/algorithms used on MPSoC. The power models of the scheduling approaches mostly do not replicate the overall computing system and results are based on assumptions. Specifically, transition, sleep, and migration overheads are not considered during energy-efficiency calculation. Furthermore, the studies on energy-aware task scheduling also ignore energy consumption due to frequency clock generation and memory. Concisely, the energy-efficiency of many approaches depends on a number of assumptions and hardware characteristics.

Most of the approaches in the literature primarily focus on reducing either communication or processing energy consumption. Some energy management approaches decrease both the processing and communication energy but do not provide a contention-aware and energy-efficient scheduling solution. One key concern is network contention which is caused by the limited number of resources within the network. This kind of contention occurs when many communications (associated with different) edges compete for the network resources. To successfully handle this kind of contention in scheduling a new, more realistic target system model for contention-aware task scheduling is required. Such a system model has been defined in (170).

1. In future as a investigation on scheduling approach could be performed to minimise both the static and dynamic energy consumption while considering communications contention at NoC links for heterogeneous MPSoC system by considering the energy performance profiles of the processors.

In MPSoCs static power dominates the total power consumption (171). Chen et al. (109) have argued that DPM and DVFS should be applied collectively rather in phases. They have proposed an MILP based approach that optimally integrates DPM and DVFS. Their approach performs significantly better than the approaches that apply DVF first and then DPM or DPM first and then DVFS. However their approach is not scalable.

1. One interesting future could be performing energy-aware scalable task mapping while integrating DPM and DVFS within the scheduling i.e. performing mapping, task ordering, voltage scaling, and static power management in an integrated way.
2. Another possible future research work could be to support VFI based MPSoC system while meeting certain constraints. One such constraint is, within one island cores must share same frequency and the other one is that cores should enter into sleep-mode and switch to active-mode at the same time.

Coarse-grained software pipelining also known as Re-timing is and effective system level energy optimisation technique when integrated with DVFS. Unfortunately, re-timing causes increased latency and memory overhead. Previous investigations (156; 143; 128; 154; 155) mainly focused on minimising the extra prologue delay but neglected the memory-overhead. Memory-overhead introduced due to re-timing is a considerable issue because in the worst case it may introduce violations of the memory capacity bounds.

1. Another problem that researcher in the future can cope with is to integrate DVFS and DPM with the re-timing technique while aiming to decrease the memory-overhead along with energy consumption reduction for dependent tasks with precedence and deadline constraints. Re-timing can be used to schedule tasks with precedence and deadline constraints on VFI based NoC-MPSoCs considering memory capacity constraints while reducing the total energy consumption and re-timing latency.

The evolutionary algorithms belong to stochastic generate and test algorithms that are based on (1) exploration of the search space and (2) exploitation of the promising information already found. Nevertheless, the search-based algorithms discussed in the literature fail to efficiently exploit the available chunk of information i.e. schemata. Moreover, exploration and exploitation are the two opposing forces, a well-found balance between exploration and exploitation determines the success of a search based algorithm.

1. There is an urgency of novel population-based algorithms for offline task scheduling on MPSoCs that can dynamically switch between explorative and exploitative search modes at run-time for performance trade-off.

Though, research (94) exploited the variations in workload by adjusting the speed dynamically for multiprocessor systems. However, this study controlled inter-VFI queue occupancy and not reducing the energy consumption of the full VFI based computing system. Similarly, the research work (172) developed a feedback control system for online fine grain voltage control in VFI based MPSoC systems.

1. These studies modelled the workloads via linear state-space models but do not include timing constraints in the problem formulation. It would be an interesting solution to analyse full VFI based MPSoC system energy performance while considering timing constraints.

Nevertheless, the study in (173) demonstrated that reinforcement learning based framework efficiently works with DVFS and different scheduling policies for a single processor per VFI MPSoCs. The framework performed voltage and frequency scaling decisions adaptable to processor configurations and task characteristics. However, in the future thermal-aware (as the technology parameters are shrinking from 90 nm to 7 nm) task scheduling on MPSoCs using DVFS would be the center of attraction for the researchers. Using this framework a thermal-aware task scheduling can be performed on MPSoCs.

1. One of the future work could be extending this existing framework by including a learning-based load balancing mechanism to manage tasks execution among the processors based on processor temperature, task characteristics, and processor architecture.
2. More parameters could be included into the framework to capture task characteristics precisely, while the task model could be extended to include various other types of tasks e.g. dependent tasks (DAGs, CTGs) and sporadic tasks.
3. Similarly an interesting future problem could be investigated using tensor train networks and tensor decomposition for improving the training efficiency double deep Q-learning model presented in (174) for energy-efficient edge scheduling.

Task scheduling optimisation minimises service response time, cost and significantly improves QoS but it is a non-linear NP-hard problem in dynamic cloud environment. Main aim of dynamic scheduling focuses on adapting inherent uncertainty and coping with conflicting objectives, for example reducing task transfer times, task queue lengths, task execution costs, response times and power consumption.

1. Evolutionary algorithms can be integrated with dynamic task schedulers in cloud environment to minimise search space complexity and guarantee acceptable/minimal run-time for the dynamic scheduling algorithms.
2. Existing task scheduling techniques in cloud computing fail to achieve an acceptable trade-off between energy consumption and application performance. Thus, there is a need of a scheduler to attain an optimised performance and energy-efficiency.
3. One of the future work could be designing an adaptive energy minimisation algorithm/scheduler for multiprocessor systems to efficiently adjust the performance and workload variations across the applications and within the application.

9. Conclusion

In this paper, we thoroughly examined Internet-of-Things (IoT), Wireless Sensor Network (WSN), and a structure of a Sensor Node (SN). We critically analysed the use of Multiprocessor System-on-Chip (MPSoC) platform in consumer electronics such as smart-phone plus IoT based computationally extensive applications and discussed MPSoC superiority over single-core processors in terms of energy conservation, performance, and size. Furthermore, we investigated MPSoCs considering important characteristics, such as energy savings capability, scalability, architecture, and communication techniques. In addition, we explored in-depth software level energy optimisation techniques i.e DVFS, DPM, and re-timing. We examined energy-aware scheduling approaches and/or algorithms used in embedded system and SN level, particularly targeting MPSoC architectures. We identified challenges to existing energy-aware approaches, algorithms and provided insightful suggestions for future research directions in energy-efficient task scheduling and mapping on MPSoCs systems. We identified technological opportunities which provided intuitive direction for further research, including, (1) inefficiency to address the problem of communication contention at the NoC links for heterogeneous MPSoC systems, (2) inadequacy to establish an optimal balance between DPM and DVFS for scalable task scheduling, (3) insufficient scheduling approaches to focus on reducing memory overhead and prologue in the re-timing integrated with DVFS and/or DPM technique (4) ineffective search based algorithms that can dynamically switch between explorative and exploitative modes at run time, (5) incapable energy-aware approaches to consider the power model of the entire computing system including memory, (6) deficiency to schedule task on processors while considering processor temperature and task constraints, etc.

References

[1] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran, and S. Guizani, "Internet-of-things-based smart cities: Recent advances and challenges," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 16–24, 2017.

[2] R. Dou and G. Nan, "Optimizing sensor network coverage and regional connectivity in industrial iot systems," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1351–1360, 2017.

[3] J. Colding and S. Barthel, "An urban ecology critique on the "smart city" model," *Journal of Cleaner Production*, vol. 164, pp. 95–101, 2017.

[4] M. De Jong, S. Joss, D. Schraven, C. Zhan, and M. Weijnen, "Sustainable–smart–resilient–low carbon–eco–knowledge cities; making sense of a multitude of concepts promoting sustainable urbanization," *Journal of Cleaner production*, vol. 109, pp. 25–38, 2015.

[5] C. Perera, C. H. Liu, S. Jayawardena, and M. Chen, "A survey on internet of things from industrial market perspective," *IEEE Access*, vol. 2, pp. 1660–1679, 2014.

[6] J. H. Kim, "A survey of iot security: Risks, requirements, trends, and key technologies," *Journal of Industrial Integration and Management*, vol. 2, no. 02, p. 1750008, 2017.

[7] C. I. ROTUNĂ, C. E. CÎRNU, and A. GHEORGHIȚĂ, "Implementing smart city solutions: Smart city map and city drop," *Quality of Life (1018-0389)/Calitatea Vietii*, vol. 28, no. 3, 2017.

[8] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[9] D. Li, J. Cao, and Y. Yao, "Big data in smart cities," *Science China Information Sciences*, vol. 58, no. 10, pp. 1–12, 2015.

[10] J. Huang, C. Buckl, A. Raabe, and A. Knoll, "Energy-aware task allocation for network-on-chip based heterogeneous multiprocessor systems," in *19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2011*. IEEE, 2011, pp. 447–454.

[11] 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*, 2018.

[12] E. Gelenbe and Y. Caseau, "The impact of information technology on energy consumption and carbon emissions," *Ubiquity*, vol. 2015, no. June, p. 1, 2015.

[13] S. Cui, A. J. Goldsmith, and A. Bahai, "Energy-efficiency of mimo and cooperative mimo techniques in sensor networks," *IEEE Journal on selected areas in communications*, vol. 22, no. 6, pp. 1089–1098, 2004.

[14] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Computer Networks*, vol. 67, pp. 104–122, 2014.

[15] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[16] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[17] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of things: Security vulnerabilities and challenges," in *Computers and Communication (ISCC), 2015 IEEE Symposium on*. IEEE, 2015, pp. 180–187.

[18] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "On the integration of cloud computing and internet of things," in *Future internet of things and cloud (FiCloud), 2014 international conference on*. IEEE, 2014, pp. 23–30.

[19] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[20] P. P. Ray, "A survey on internet of things architectures," *Journal of King Saud University-Computer and Information Sciences*, vol. 30, no. 3, pp. 291–319, 2018.

[21] F. K. Shaikh, S. Zeadally, and E. Exposito, "Enabling technologies for green internet of things," *IEEE Systems Journal*, vol. 11, no. 2, pp. 983–994, 2017.

[22] R. V. Kulkarni and G. K. Venayagamoorthy, "Particle swarm optimization in wireless-sensor networks: A brief survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 2, pp. 262–267, 2010.

[23] K. Islam, W. Shen, and X. Wang, "Wireless sensor network reliability and security in factory automation: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1243–1256, 2012.

[24] H. Yetgin, K. T. K. Cheung, M. El-Hajjar, and L. H. Hanzo, "A survey of network lifetime maximization techniques in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 828–854, 2017.

[25] S. Mittal, "A survey of techniques for improving energy efficiency in embedded computing systems," *International Journal of Computer Aided Engineering and Technology*, vol. 6, no. 4, pp. 440–459, 2014.

[26] F. Siddiqui, M. Hagan, and S. Sezer, "Embedded policing and policy enforcement approach for future secure iot technologies," 2018.

[27] M. Bambagini, M. Marinoni, H. Aydin, and G. Buttazzo, "Energy-aware scheduling for real-time systems: a survey," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 15, no. 1, p. 7, 2016.

[28] S. . K. C. Kumar and Vijay, "A strategy for elimination of data redundancy in internet of things (iot) based wireless sensor network (wsn)," pp. 1–9, 2018.

[29] Z. Zhou, D. Zhao, L. Liu, and P. C. Hung, "Energy-aware composition for wireless sensor networks as a service," *Future Generation Computer Systems*, vol. 80, pp. 299–310, 2018.

[30] C. Alcaraz, P. Najera, J. Lopez, and R. Roman, "Wireless sensor networks and the internet of things: Do we need a complete integration?" in *1st International Workshop on the Security of the Internet of Things (SecIoT'10)*, 2010.

[31] I. Lee and K. Lee, "The internet of things (iot): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[32] É. L. Souza, E. F. Nakamura, and R. W. Pazzi, "Target tracking for sensor networks: A survey," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, p. 30, 2016.

[33] D. C. Harrison, W. K. Seah, and R. Rayudu, "Rare event detection and propagation in wireless sensor networks," *ACM Computing Surveys (CSUR)*, vol. 48, no. 4, p. 58, 2016.

[34] D. Kandris, M. Tsagkaropoulos, I. Politis, A. Tzes, and S. Kotsopoulos, "Energy efficient and perceived qos aware video routing over

wireless multimedia sensor networks," *Ad Hoc Networks*, vol. 9, no. 4, pp. 591–607, 2011.

[35] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer networks*, vol. 51, no. 4, pp. 921–960, 2007.

[36] Z.-J. Zhang, C.-F. Lai, and H.-C. Chao, "A green data transmission mechanism for wireless multimedia sensor networks using information fusion," *IEEE Wireless Communications*, vol. 21, no. 4, pp. 14–19, 2014.

[37] L.-m. Ang, K. P. Seng, L. W. Chew, L. S. Yeong, and W. C. Chia, "Wireless multimedia sensor network technology," in *Wireless multimedia sensor networks on reconfigurable hardware*. Springer, 2013, pp. 5–38.

[38] L. Zhao, P. Pop, Z. Zheng, and Q. Li, "Timing analysis of avb traffic in tsn networks using network calculus," in *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2018, pp. 25–36.

[39] M. Stopper and B. Katalinic, "Service-oriented architecture design aspects of opc ua for industrial applications," in *Proceedings of the International Multi-Conference of Engineers and Computer Scientists*, vol. 2. Citeseer, 2009.

[40] F. Karray, M. W. Jmal, M. Abid, M. S. BenSaleh, and A. M. Obeid, "A review on wireless sensor node architectures," in *2014 9th International Symposium on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*. IEEE, 2014, pp. 1–8.

[41] S. F. Johann, M. T. Moreira, L. S. Heck, N. L. Calazans, and F. P. Hessel, "A processor for iot applications: An assessment of design space and trade-offs," *Microprocessors and Microsystems*, vol. 42, pp. 156–164, 2016.

[42] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[43] J. Ceng and R. Leupers, "A methodology for efficient multiprocessor system on chip software development," Lehr-und Forschungsgebiet Software für Systeme auf Silizium, Tech. Rep., 2011.

[44] M. M. Waldrop, "The chips are down for moore's law," *Nature News*, vol. 530, no. 7589, p. 144, 2016.

[45] V. Rajaraman, "Multi-core microprocessors," *Resonance*, vol. 22, no. 12, pp. 1175–1192, 2017.

[46] W. Wolf, "The future of multiprocessor systems-on-chips," in *Proceedings of the 41st annual Design Automation Conference*. ACM, 2004, pp. 681–685.

[47] A. A. Jerraya and W. Wolf, "The what, why, and how of mpsocs," *Multiprocessor Systems-on-Chips*, pp. 1–18, 2005.

[48] S. Mishra, N. K. Singh, and V. Rousseau, *System on Chip Interfaces for Low Power Design*. Morgan Kaufmann, 2015.

[49] M. Bohr, "The new era of scaling in an soc world," in *Solid-State Circuits Conference-Digest of Technical Papers, 2009. ISSCC 2009. IEEE International*. IEEE, 2009, pp. 23–28.

[50] E. W. Wachter, G. V. Merrett, B. M. Al-Hashimi, and A. K. Singh, "Reliable mapping and partitioning of performance-constrained opencl applications on cpu-gpu mpsocs," in *Proceedings of the 15th IEEE/ACM Symposium on Embedded Systems for Real-Time Multimedia*. ACM, 2017, pp. 78–83.

[51] T. A. Claasen, "An industry perspective on current and future state of the art in system-on-chip (soc) technology," *Proceedings of the IEEE*, vol. 94, no. 6, pp. 1121–1137, 2006.

[52] L. Chen, N. Boichat, and T. Mitra, "Customized mpsoc synthesis for task sequence," in *Application Specific Processors (SASP), 2011 IEEE 9th Symposium on*. IEEE, 2011, pp. 16–21.

[53] L. Yan, L. Renfa, X. Cheng, and Y. Fei, "Hw-sw framework for multimedia applications on mpsoc: practice and experience," *Journal of computers*, vol. 4, no. 3, pp. 238–244, 2009.

[54] Y. Sasagawa and A. Mori, "High-level video analytics pc subsystem using soc with heterogeneous multicore architecture," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 4, pp. 1051–1059, 2016.

[55] M. Magno, F. Tombari, D. Brunelli, L. Di Stefano, and L. Benini, "Multimodal video analysis on self-powered resource-limited wireless smart camera," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 3, no. 2, pp. 223–235, 2013.

[56] S. Saponara, L. Fanucci, and E. Petri, "A multi-processor noc-based architecture for real-time image/video enhancement," *Journal of Real-Time Image Processing*, vol. 8, no. 1, pp. 111–125, 2013.

[57] 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 of Things Journal*, vol. 5, no. 3, pp. 1598–1605, 2018.

[58] A. Safaei, Q. J. Wu, and Y. Yang, "System-on-a-chip (soc)-based hardware acceleration for foreground and background identification," *Journal of the Franklin Institute*, vol. 355, no. 4, pp. 1888–1912, 2018.

[59] H. Meng, M. Freeman, N. Pears, and C. Bailey, "Real-time human action recognition on an embedded, reconfigurable video processing architecture," *Journal of Real-Time Image Processing*, vol. 3, no. 3, pp. 163–176, 2008.

[60] A. A. Abbo, R. P. Kleihorst, and B. Schueler, "Xetal-ii: A low-power massively-parallel processor for video scene analysis," *Journal of Signal Processing Systems*, vol. 62, no. 1, pp. 17–27, 2011.

[61] F. Karray, W. M. Jmal, M. Abid, D. Houssaini, A. M. Obeid, S. M. Qasim, and M. S. BenSaleh, "Architecture of wireless sensor nodes for water monitoring applications: From microcontroller-based system to soc solutions," in *Environmental Instrumentation and Measurements (IMEKO), 2014 5th IMEKO TC19 Symposium on*, 2014, pp. 20–24.

[62] 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.

[63] I. A. Khatib, F. Poletti, D. Bertozzi, L. Benini, M. Bechara, H. Khalifeh, A. Jantsch, and R. Nabiev, "A multiprocessor system-on-chip for real-time biomedical monitoring and analysis: Ecg prototype architectural design space exploration," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 13, no. 2, p. 31, 2008.

[64] G. Kavya and V. ThulasiBai, "Wearable advanced single chip ecg telemonitoring system using sopc," *IEICE Electronics Express*, vol. 11, no. 6, pp. 20 140 097–20 140 097, 2014.

[65] A. Iranfar, A. Pahlevan, M. Zapater, M. Žagar, M. Kovač, and D. Atienza, "Online efficient bio-medical video transcoding on mpsocs through content-aware workload allocation," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 949–954.

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

[67] A. B. Ahmed, Y. Kimezawa, and A. B. Abdallah, "Towards smart health monitoring system for elderly people," in *4th International Conference on Awareness Science and Technology*. IEEE, 2012, pp. 248–253.

[68] J. Álvarez-Bermejo, D. Morales-Santos, E. Castillo-Morales, L. Parrilla, and J. López-Ramos, "Efficient image-based analysis of fruit surfaces using ccd cameras and smartphones," *The Journal of Supercomputing*, pp. 1–12, 2018.

[69] N. Jangid and B. Sharma, "Cloud computing and robotics for disaster management," in *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*. IEEE, 2016, pp. 20–24.

[70] S. Niar, A. Yurdakul, O. Unsal, T. Tugcu, and A. Yuceturk, "A dynamically reconfigurable architecture for emergency and disaster management in its," in *2014 International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE, 2014, pp. 479–484.

[71] C. Tran, "Structural-damage detection with big data using parallel computing based on mpsoc," *International Journal of Machine Learning and Cybernetics*, vol. 7, no. 6, pp. 1213–1223, 2016.

[72] M. Hassan, "Heterogeneous mpsocs for mixed criticality systems: Challenges and opportunities," *IEEE Design & Test*, 2017.

[73] H. Youness, M. Moness, and M. Khaled, "Mpsocs and multicore

microcontrollers for embedded pid control: A detailed study," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2122–2134, 2014.

[74] A. Abraham and S. Das, *Computational intelligence in power engineering*. Springer, 2010, vol. 302.

[75] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A survey on visual content-based video indexing and retrieval," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 797–819, 2011.

[76] A. Gandomi and M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, vol. 35, no. 2, pp. 137–144, 2015.

[77] T. Dorta, J. Jiménez, J. L. Martín, U. Bidarte, and A. Astarloa, "Overview of fpga-based multiprocessor systems," in *2009 International Conference on Reconfigurable Computing and FPGAs*. IEEE, 2009, pp. 273–278.

[78] Z. Wang, W. Liu, J. Xu, B. Li, R. Iyer, R. Illikkal, X. Wu, W. H. Mow, and W. Ye, "A case study on the communication and computation behaviors of real applications in noc-based mpsocs," in *2014 IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2014, pp. 480–485.

[79] 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*. Springer, 2011, pp. 1–21.

[80] A. Pathania, A. E. Irimiea, A. Prakash, and T. Mitra, "Power-performance modelling of mobile gaming workloads on heterogeneous mpsocs," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2015, pp. 1–6.

[81] E. Carvalho, N. Calazans, and F. Moraes, "Heuristics for dynamic task mapping in noc-based heterogeneous mpsocs," in *Rapid System Prototyping, 2007. RSP 2007. 18th IEEE/IFIP International Workshop on*. IEEE, 2007, pp. 34–40.

[82] L. Benini and G. De Micheli, "Networks on chips: A new paradigm for component-based mpsoc design," *Proc. MPSoC*, 2004.

[83] E. Salminen, V. Lahtinen, K. Kuusilinna, and T. Hamalainen, "Overview of bus-based system-on-chip interconnections," in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, vol. 2. IEEE, 2002, pp. II–II.

[84] S. Pasricha, N. D. Dutt, and M. Ben-Romdhane, "Bmsyn: Bus matrix communication architecture synthesis for mpsoc," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 8, pp. 1454–1464, 2007.

[85] P. Gandhani, "Moving from amba ahb to axi bus in soc designs: A comparative study," *International Journal of Computer Science & Emerging Technologies*, vol. 2, no. 4, 2011.

[86] A. AMBA, "Axi and ace™ protocol specification," 2011.

[87] U. Y. Ogras, R. Marculescu, and D. Marculescu, "Variation-adaptive feedback control for networks-on-chip with multiple clock domains," in *Proceedings of the 45th annual Design Automation Conference*. ACM, 2008, pp. 614–619.

[88] H. Gu, J. Xu, and W. Zhang, "A low-power fat tree-based optical network-on-chip for multiprocessor system-on-chip," in *Proceedings of the conference on Design, Automation and Test in Europe*. European Design and Automation Association, 2009, pp. 3–8.

[89] 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, 2015.

[90] S. Hesham, J. Rettkowski, D. Goehringer, and M. A. A. El Ghany, "Survey on real-time networks-on-chip," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1500–1517, 2017.

[91] Tilera. (2019) SoCs tilera-gx. [Online]. Available: http://http://www.mellanox.com/page/products_dyn?product_family=238&mtag=tile_gx72

[92] T. Corporation. (2019) SoCs tilera-gx. [Online]. Available: http://www.design-reuse.com/articles/27489/multi-fpga-noc-based-64-core-mpsoc.html

[93] 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.

[94] S. Garg, D. Marculescu, R. Marculescu, and U. Ogras, "Technology-driven limits on dvfs controllability of multiple voltage-frequency island designs: a system-level perspective," in *Proceedings of the 46th Annual Design Automation Conference*. ACM, 2009, pp. 818–821.

[95] P. P. Pande, A. Ganguly, and K. Chakrabarty, *Design Technologies for Green and Sustainable Computing Systems*. Springer, 2013.

[96] J.-J. Han, X. Wu, D. Zhu, H. Jin, L. T. Yang, and J.-L. Gaudiot, "Synchronization-aware energy management for vfi-based multicore real-time systems," *IEEE Transactions on Computers*, vol. 61, no. 12, pp. 1682–1696, 2012.

[97] M. E. Gerards, J. L. Hurink, and J. Kuper, "On the interplay between global dvfs and scheduling tasks with precedence constraints," *IEEE Transactions on Computers*, vol. 64, no. 6, pp. 1742–1754, 2015.

[98] M. C. Calzarossa, L. Massari, and D. Tessera, "Workload characterization: A survey revisited," *ACM Computing Surveys (CSUR)*, vol. 48, no. 3, p. 48, 2016.

[99] A. Andreev, F. Kaplan, M. Zapater, A. K. Coskun, and D. Atienza, "Design optimization of 3d multi-processor system-on-chip with integrated flow cell arrays," 2018.

[100] K. P. B. P. Banerjee, "An approximate algorithm for the partitionable independent task scheduling problem," *Urbana*, vol. 51, p. 61801, 1990.

[101] K. Cao, J. Zhou, M. Yin, T. Wei, and M. Chen, "Static thermal-aware task assignment and scheduling for makespan minimization in heterogeneous real-time mpsocs," in *System and Software Reliability (ISSSR), International Symposium on*. IEEE, 2016, pp. 111–118.

[102] T. Wei, J. Zhou, K. Cao, P. Cong, M. Chen, G. Zhang, X. S. Hu, and J. Yan, "Cost-constrained qos optimization for approximate computation real-time tasks in heterogeneous mpsocs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2017.

[103] Y.-K. Kwok and I. Ahmad, "Static scheduling algorithms for allocating directed task graphs to multiprocessors," *ACM Computing Surveys (CSUR)*, vol. 31, no. 4, pp. 406–471, 1999.

[104] H. I. Ali, B. Akesson, and L. M. Pinho, "Combining dataflow applications and real-time task sets on multi-core platforms," in *Proceedings of the 20th International Workshop on Software and Compilers for Embedded Systems*. ACM, 2017, pp. 60–63.

[105] H. Arabnejad and J. G. Barbosa, "List scheduling algorithm for heterogeneous systems by an optimistic cost table," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 682–694, 2014.

[106] L. Huang and Q. Xu, "Performance yield-driven task allocation and scheduling for mpsocs under process variation," in *Design Automation Conference (DAC), 2010 47th ACM/IEEE*. IEEE, 2010, pp. 326–331.

[107] J. Zhou, J. Yan, K. Cao, Y. Tan, T. Wei, M. Chen, G. Zhang, X. Chen, and S. Hu, "Thermal-aware correlated two-level scheduling of real-time tasks with reduced processor energy on heterogeneous mpsocs," *Journal of Systems Architecture*, vol. 82, pp. 1–11, 2018.

[108] 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.

[109] G. Chen, K. Huang, and A. Knoll, "Energy optimization for real-time multiprocessor system-on-chip with optimal dvfs and dpm combination," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 13, no. 3s, p. 111, 2014.

[110] 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.

[111] H. Topcuoglu, S. Hariri, and M.-y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE transactions on parallel and distributed systems*, vol. 13, no. 3,

pp. 260–274, 2002.

[112] U. U. Tariq and H. Wu, "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.

[113] 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.

[114] P. Krömer, J. Platoš, V. Snášel, and A. Abraham, "A comparison of many-threaded differential evolution and genetic algorithms on cuda," in *2011 Third World Congress on Nature and Biologically Inspired Computing*. IEEE, 2011, pp. 509–514.

[115] H. Orsila, "Optimizing algorithms for task graph mapping on multiprocessor system on chip," *Tampereen teknillinen yliopisto. Julkaisu-Tampere University of Technology. Publication*, vol. 972, 2011.

[116] W. Zhang, E. Bai, H. He, and A. M. Cheng, "Solving energy-aware real-time tasks scheduling problem with shuffled frog leaping algorithm on heterogeneous platforms," *Sensors*, vol. 15, no. 6, pp. 13 778–13 804, 2015.

[117] N. Kumar and D. P. Vidyarthi, "A ga based energy aware scheduler for dvfs enabled multicore systems," *Computing*, pp. 1–23, 2017.

[118] 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 *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*. IEEE, 2014, pp. 1–6.

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

[120] A. K. Coskun, T. S. Rosing, and K. Whisnant, "Temperature aware task scheduling in mpsocs," in *Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE'07*. IEEE, 2007, pp. 1–6.

[121] E. Le Sueur and G. Heiser, "Dynamic voltage and frequency scaling: The laws of diminishing returns," in *Proceedings of the 2010 international conference on Power aware computing and systems*, 2010, pp. 1–8.

[122] S. Mittal and J. S. Vetter, "A survey of cpu-gpu heterogeneous computing techniques," *ACM Computing Surveys (CSUR)*, vol. 47, no. 4, p. 69, 2015.

[123] T. R. da Rosa, V. Larréa, N. Calazans, and F. G. Moraes, "Power consumption reduction in mpsocs through dfs," in *2012 25th Symposium on Integrated Circuits and Systems Design (SBCCI)*. IEEE, 2012, pp. 1–6.

[124] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proceedings of the 2007 international symposium on Low power electronics and design (ISLPED'07)*. IEEE, 2007, pp. 38–43.

[125] Z. Najam, M. Y. Qadri, and S. Najam, "Real-time implementation of dvfs enhanced leon3 mpsoc on fpga," in *Intelligent and Advanced Systems (ICIAS), 2016 6th International Conference on*. IEEE, 2016, pp. 1–6.

[126] C. Poellabauer, L. Singleton, and K. Schwan, "Feedback-based dynamic voltage and frequency scaling for memory-bound real-time applications," in *Real Time and Embedded Technology and Applications Symposium, 2005. RTAS 2005. 11th IEEE*. IEEE, 2005, pp. 234–243.

[127] A. Das, M. J. Walker, A. Hansson, B. M. Al-Hashimi, and G. V. Merrett, "Hardware-software interaction for run-time power optimization: A case study of embedded linux on multicore smartphones," in *2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2015, pp. 165–170.

[128] 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.

[129] 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.

[130] D. Zhu, R. Melhem, and B. R. Childers, "Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 7, pp. 686–700, 2003.

[131] 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.

[132] 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.

[133] 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, 2015.

[134] A. Singh, A. Prakash, K. R. Basireddy, G. Merrett, and B. Al-Hashimi, "Energy efficient run-time mapping and thread partitioning of concurrent opencl applications on cpu-gpu mpsocs," *ACM Transactions on Embedded Computing Systems*, 2017.

[135] F. Gruian, "System-level design methods for low-energy architectures containing variable voltage processors," in *International Workshop on Power-Aware Computer Systems*. Springer, 2000, pp. 1–12.

[136] Y. C. Lee and A. Y. Zomaya, "Energy conscious scheduling for distributed computing systems under different operating conditions," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1374–1381, 2011.

[137] J. Kołodziej, S. U. Khan, L. Wang, and A. Y. Zomaya, "Energy efficient genetic-based schedulers in computational grids," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 4, pp. 809–829, 2015.

[138] T. Maqsood, S. Ali, S. U. Malik, and S. A. Madani, "Dynamic task mapping for network-on-chip based systems," *Journal of Systems Architecture*, vol. 61, no. 7, pp. 293–306, 2015.

[139] N. Chatterjee, S. Paul, P. Mukherjee, and S. Chattopadhyay, "Deadline and energy aware dynamic task mapping and scheduling for network-on-chip based multi-core platform," *Journal of Systems Architecture*, vol. 74, pp. 61–77, 2017.

[140] N. Chatterjee, S. Paul, and S. Chattopadhyay, "Task mapping and scheduling for network-on-chip based multi-core platform with transient faults," *Journal of Systems Architecture*, 2018.

[141] M. K. Bhatti, C. Belleudy, and M. Auguin, "Hybrid power management in real time embedded systems: an interplay of dvfs and dpm techniques," *Real-Time Systems*, vol. 47, no. 2, pp. 143–162, 2011.

[142] P. Ghosh, A. Sen, and A. Hall, "Energy efficient application mapping to noc processing elements operating at multiple voltage levels," in *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*. IEEE Computer Society, 2009, pp. 80–85.

[143] 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.

[144] 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.

[145] M. Abdel-Basset, D. El-Shahat, K. Deb, and M. Abouhawwash, "Energy-aware whale optimization algorithm for real-time task scheduling in multiprocessor systems," *Applied Soft Computing*, vol. 93, p. 106349, 2020.

[146] Z. Deng, D. Cao, H. Shen, Z. Yan, and H. Huang, "Reliability-aware task scheduling for energy efficiency on heterogeneous multiprocessor systems," *The Journal of Supercomputing*, pp. 1–39, 2021.

[147] A. Andrei, M. Schmitz, P. Eles, Z. Peng, and B. M. Al Hashimi, "Simultaneous communication and processor voltage scaling for dynamic and leakage energy reduction in time-constrained systems,"

in *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*. IEEE Computer Society, 2004, pp. 362–369.

[148] 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.

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

[150] D. Li and J. Wu, "Energy-efficient contention-aware application mapping and scheduling on noc-based mpsocs," *Journal of Parallel and Distributed Computing*, vol. 96, pp. 1–11, 2016.

[151] 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, p. 66, 2019.

[152] S. Ninomiya, K. Sakanushi, Y. Takeuchi, and M. Imai, "Task allocation and scheduling for voltage-frequency islands applied noc-based mpsoc considering network congestion," in *Embedded Multicore Socs (MCSoC), 2012 IEEE 6th International Symposium on*. IEEE, 2012, pp. 107–112.

[153] 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*, vol. 111, pp. 1–16, 2020.

[154] 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.

[155] H. Liu, Z. Shao, M. Wang, J. Du, C. J. Xue, and Z. Jia, "Combining coarse-grained software pipelining with dvs for scheduling real-time periodic dependent tasks on multi-core embedded systems," *Journal of Signal Processing Systems*, vol. 57, no. 2, pp. 249–262, 2009.

[156] Z. Shao, M. Wang, Y. Chen, C. Xue, M. Qiu, L. T. Yang, and E. H.-M. Sha, "Real-time dynamic voltage loop scheduling for multi-core embedded systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, no. 5, pp. 445–449, 2007.

[157] U. U. Tariq, H. Ali, L. Liu, J. Hardy, M. Kazim, and W. Ahmed, "Energy-aware scheduling of streaming applications on edge-devices in iot based healthcare," *IEEE Transactions on Green Communications and Networking*, 2021.

[158] N. S. Kim, T. Kgil, K. Bowman, V. De, and T. Mudge, "Total power-optimal pipelining and parallel processing under process variations in nanometer technology," in *Computer-Aided Design, 2005. ICCAD-2005. IEEE/ACM International Conference on*. IEEE, 2005, pp. 535–540.

[159] K. Bhatti, C. Belleudy, and M. Auguin, "Power management in real time embedded systems through online and adaptive interplay of dpm and dvfs policies," in *2010 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*. IEEE, 2010, pp. 184–191.

[160] D.-C. Juan and D. Marculescu, "Power-aware performance increase via core/uncore reinforcement control for chip-multiprocessors," in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*, 2012, pp. 97–102.

[161] H. Shen, Y. Tan, J. Lu, Q. Wu, and Q. Qiu, "Achieving autonomous power management using reinforcement learning," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 18, no. 2, pp. 1–32, 2013.

[162] M. I. Khan and B. Rinner, "Energy-aware task scheduling in wireless sensor networks based on cooperative reinforcement learning," in *2014 IEEE International Conference on Communications Workshops (ICC)*. IEEE, 2014, pp. 871–877.

[163] N. K. Pham, A. Kumar, A. K. Singh, and M. M. A. Khin, "Leakage aware resource management approach with machine learning optimization framework for partially reconfigurable architectures," *Microprocessors and Microsystems*, vol. 47, pp. 231–243, 2016.

[164] A. Dambreville, J. Tomasik, J. Cohen, and F. Dufoulon, "Load prediction for energy-aware scheduling for cloud computing platforms," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2604–2607.

[165] M. I. Khan, K. Xia, A. Ali, and N. Aslam, "Energy-aware task scheduling by a true online reinforcement learning in wireless sensor networks," *International Journal of Sensor Networks*, vol. 25, no. 4, pp. 244–258, 2017.

[166] H. M. Makrani, H. Sayadi, D. Motwani, H. Wang, S. Rafatirad, and H. Homayoun, "Energy-aware and machine learning-based resource provisioning of in-memory analytics on cloud," in *Proceedings of the ACM Symposium on Cloud Computing*, 2018, pp. 517–517.

[167] A. Esmaili and M. Pedram, "Energy-aware scheduling of jobs in heterogeneous cluster systems using deep reinforcement learning," in *2020 21st International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2020, pp. 426–431.

[168] Z. Yu, P. Machado, A. Zahid, A. M. Abdulghani, K. Dashtipour, H. Heidari, M. A. Imran, and Q. H. Abbasi, "Energy and performance trade-off optimization in heterogeneous computing via reinforcement learning," *Electronics*, vol. 9, no. 11, p. 1812, 2020.

[169] Y. Qin, H. Wang, S. Yi, X. Li, and L. Zhai, "An energy-aware scheduling algorithm for budget-constrained scientific workflows based on multi-objective reinforcement learning," *The Journal of Supercomputing*, vol. 76, no. 1, pp. 455–480, 2020.

[170] 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.

[171] R. Jejurikar, C. Pereira, and R. Gupta, "Leakage aware dynamic voltage scaling for real-time embedded systems," in *Proceedings of the 41st annual Design Automation Conference*. ACM, 2004, pp. 275–280.

[172] 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.

[173] F. M. M. ul Islam, M. Lin, L. T. Yang, and K.-K. R. Choo, "Task aware hybrid dvfs for multi-core real-time systems using machine learning," *Information Sciences*, vol. 433, pp. 315–332, 2018.

[174] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, S. U. Khan, and P. Li, "A double deep q-learning model for energy-efficient edge scheduling," *IEEE Transactions on Services Computing*, 2018.