Journal of Circuits, Systems, and Computers
Vol. 28, No. 7 (2019) 1930007 (25 pages)

DOI: 10.1142/S0218126619300071

A Review of Recent Techniques in Mixed-Criticality Systems*

Hongxia Chai†, Gongxuan Zhang†, Jin Sun†,
Ahmadreza Vajdi†, Jing Hua‡ and Junlong Zhou†,§

†*School of Computer Science and Engineering,*
Nanjing University of Science and Technology,
200 Xiaolingwei Street, Nanjing,
Jiangsu 210094, P. R. China

‡*School of Software,*
Jiangxi Agricultural University,
Nanchang 330045, P. R. China
§*jlzhou@njust.edu.cn*

Received 12 March 2018
Accepted 26 July 2018
Published 5 September 2018

Unlike traditional embedded systems that almost have only one criticality level, many complex embedded systems nowadays are mixed-critical and are more and more widely used. There has been a lot of research on mixed-criticality (MC) systems. In this paper, we present a survey on the MC systems on these research. First, we discuss the exaltation of the schedulability of MC systems. As improving schedulability may lead to quality-of-service (QoS) reduction of MS systems. Therefore, we investigate the approaches to solve this problem. Improving QoS of MS systems may inevitably increase the energy consumption. Then, we introduce the researches that take the energy efficiency as a design requirement of MC systems. Few MC systems regard fault-tolerance as the design requirement, thus, we extensively investigate fault-tolerance of MC systems. In addition, we introduce some of the main applications for MC systems.

Keywords: Mixed-criticality systems; schedulability; quality of service; energy-efficiency; fault-tolerance.

1. Introduction

As the development of science and technology, the scale and complexity of the integrated functionalities in modern embedded systems also show a rapid increase, resulting in increased costs of embedded systems. To save costs, there has been a trend towards integrating applications with different key functionalities deployed in

*This paper was recommended by Regional Editor Tongquan Wei.
§Corresponding author.

an independent subsystem into a common hardware platform for sharing processor resources, which promotes the emergence of mixed-criticality (MC) systems. In order to meet the increasingly complicated tendency and the constraints of the embedded system hardware such as the size, function and energy consumption, integrating multiple critical functions deployed in independent subsystems into a unified platform has become the trend and direction of today's embedded real-time system design. Therefore, the study of MC system has become a hotpot.

MC systems have two or more different criticality levels. Here, tasks are divided according to the degree of urgency. If the tasks are more critical, they are classified as high-criticality (HC) tasks, and if the tasks are less critical, they are classified as low-criticality (LC) tasks. In MC systems, more important functionalities should be given a higher criticality to ensure safety needs for these functionalities. And in MC systems, failure to perform HC tasks can have catastrophic consequences, while failure to perform LC tasks can only result in a reduced user experience. So we need to provide varying degrees of protection to different key-level applications in MC systems. If all tasks perform within their deadlines, the system is called MC schedulable. There are many studies on schedulability in MC systems.[1–17] Most of them guarantee the accurate operation of the HC tasks, while reducing or even abandoning the execution of LC tasks. However, this will result in a reduction in the quality-of-service (QoS) of the systems, so it is unreasonable to neglect LC tasks directly. Considering this, researchers began to investigate the implementation of LC tasks to improve the QoS of the systems.[18–33] At the same time, improving system QoS will bring more energy costs. However, the energy of many MC systems is battery-powered, resulting in limited energy of the MC systems. More energy costs severely restrict the performance of embedded devices. Therefore, the application model is abstracted according to the specific application scenarios, and the correct execution of HC tasks, real-time embedded systems and low power consumption constraints are established. Under the premise of ensuring the correct implementation of the HC tasks in MC system and the real-time performance of the embedded system, finding a suitable low-power scheduling strategy is very meaningful. Based on this, energy needs to be considered when designing and developing a MC system.[34–41] The above studies have made tremendous efforts to develop techniques for dealing with the uncertainty of schedulability, QoS and energy consumption of MC systems. However, few papers regard fault tolerance as a design requirement, which represents the resilience of the system when an error occurs. Therefore, researching on fault tolerance is necessary for MC systems to enhance their robustness. Fault-tolerance is necessary for developing such systems to counter potential failures for a high level of safety and reliability. Recently, there are a lot of researches on fault tolerance in MC systems.[42–51] With the rapid development of MC system, it has been applied in more and more fields.[52–61]

Contribution and organization: This paper outlines a number of state-of-art technologies for MC systems. The test of the paper is organized as follows. First,

we discuss the importance of researching MC systems and describe the advantages, prospect, present situation of MC systems (Sec. 1) and then we present the schedulability study of MC system (Sec. 2). Next, considering that the designing of the system schedulability may result in a degradation of QoS, we probe some methods to solve this problem in MC systems (Sec. 3). However, enhancing the QoS in the system may lead to excessive energy consumption, we discuss some techniques to ensure energy efficiency (Sec. 4). In addition, we introduce the researches on fault tolerance which is taken as a requirement for the system design (Sec. 5). Finally, we introduce some applications of MC systems in different domains (Sec. 6). For a better understanding of the paper, we list the main abbreviations in Table 1.

Table 1. Main abbreviations used in the paper.

Abbreviation	Definition
MC	Mixed-criticality
QoS	Quality-of-service
HC	High-criticality
LC	Low-criticality
WCET	Worst case execution time
EDF	Earliest deadline first
VD	Virtual deadlines
AMC	Adaptive mixed-criticality
CA-TPA	Criticality-aware task partitioning algorithm
FP	Fixed priority
HPA	Heuristic priority assignment
OPA	Optimal priority assignment
DC	Dual-criticality
ZS	Zero-slack
ZSRM	Zero-slack rate-monotonic method
RBF	Request bound function
WCRT	Worst case response times
CRPD	Cache-related preemption delay
LLC	Last-level cache
RTS	Real-time system
ER	Early-release
HRT	Hard real-time
SRT	Soft real-time
OS	Operating system
IPC	Inter-process communication
GAMT	Globally Arbitrated Memory Tree
NoC	Networks-on-Chip
IoT	Internet of things
SDP	Systems development platform
FTMC	Fault-tolerant MC
MPSoCs	Multiprocessor systems-on-chip
EPT	Extended page tables
CPS	Cyber-physical systems

2. Schedulability-Aware Techniques in MC Systems

The tasks in MC system are divided into HC and LC task as mentioned above. Most papers assume that in MC systems, the worst case execution time (WCET) for HC tasks is low-level system assurance and high-level WCET is high-level system assurance, while LC tasks require only a low-level assurance, as they own a low-level WCET. During the operation, if any HC task exceeds its low-level WCET, all LC tasks will be abandoned for the system to leave sufficient system resources for satisfying the high-level assurance. If all tasks are executed within their deadline, and all tasks are performed at a low-level, and the HC tasks can satisfy their deadlines even when performing low-level WCET, the system is referred to as MC schedulable. Schedulability of tasks is the key of the MC system. To meet the schedulability of the system, most MC systems ensure the accurate operation of HC tasks, neglect or even abandon the performance of LC tasks. We will introduce the current research status below.

Many researches have been devoted to use the EDF-VD scheduling algorithm for maintaining schedulability of MC systems. Baruah *et al.*[1] considered the scheduling of implicit-deadline sporadic task in MC systems on uniform multiprocessor platforms and introduced two policies: the MC scheduling algorithm earliest deadline first-virtual deadlines (EDF-VD)[62] on uniprocessor, and the EDF-based global scheduling algorithm called fpEDF[63] on multiprocessor. Sigrist *et al.*[2] empirically assessed the influence of runtime overhead to schedulability of MC systems, and determined alternative solutions for the achievement of common MC mechanisms for multicores, and introduced two scheduling policies, one is the FTTS,[64] another is the partitioned EDF-VD.[62] Baruah[3] studied the scheduling of task in MC systems specified based on the three-parameter model of sporadic tasks on preemptive uniprocessor platforms and extended the EDF-VD scheduling algorithm[65] that schedules sporadic task in MC systems where multiple values are just used for the WCET parameter. Ramanathan and Easwaran[4] focused on the issue of MC scheduling on partitioned multiprocessor and used three different scheduling approaches ECDF,[63] EDF-VD,[62] and Adaptive MC (AMC)[65] to decrease the maximum difference between the overall LC utilization and overall HC utilization assigned on each processor. However, from the conditions of schedulability of the EDF-VD scheduler,[66] utilizations of MC tasks at other criticality levels also play an important role besides maximizing utilization. Han *et al.*[5] used the EDF-VD algorithm[62] on each core and proposed a novel criticality-aware task partitioning algorithm for a group of periodic MC tasks operating on multicore systems.

According to the synchrony hypothesis, synchronous languages asserted that all tasks must be accomplished immediately at each step of logical time. However, this assertion is improper for the design of MC systems, in which tasks like the LC tasks can endure missed deadlines. Yip *et al.*[6] proposed a new extension to the synchronous method for standing by three levels (life, noncritical and mission) of task criticality.

They achieved this by relaxing the synchrony hypothesis to allow tasks that can tolerate unbounded or bounded deadline misses. But, the scheme that relaxes the synchrony hypothesis to allow MC goes against the communication synchronous model. To solve this, they used a common lossless buffering method with limited queue sizes. Then they proposed a lossless communication model to allow mission critical tasks and life to communicate at a certain frequency. To schedule life and mission critical tasks, they developed a static scheduling approach by using integer linear programming, which maximizes system utilization by assigning slack time across all tasks proportionally. To further increase runtime utilization, the tasks will be scheduled dynamically whenever possible during the slack. Extensive results showed that the proposed approach can schedule more task sets and achieve better system utilization compared to the ER-EDF[67] approach.

Some researches used the fixed priority (FP) scheduling algorithm to maintain schedulability of MC systems. Guan *et al.*[7] proposed an effective algorithm named PLRS to schedule certifiable MC sporadic tasks systems. PLRS applies FP-jobs scheduling methods, and allocates job priorities by balancing and exploring the asymmetric influences between the workload on distinct criticality levels. Chen *et al.*[8] studied FP scheduling of MC systems on a uniprocessor platform in a more general way that uses different priority sequences in different stages of execution, and presented a novel priority assignment scheme called heuristic priority assignment (HPA) based on the popular optimal priority assignment (OPA) algorithm.[65] Hu *et al.*[9] analyzed the schedulability for dual-criticality (DC) systems with arbitrarily activated tasks. They extended the sporadic and activated task model to any activated task model in MC systems with the preemptive fixed priority tasks schedule. By applying the arrival curve to express the upper limit of task activations, they integrated the perfect consequences from RTC[68] to analyze the schedulability of any activated tasks in MC systems. However, the scheduling and analysis of MC system using sporadic MC task models have been censured for its limited suitability to many real systems. To solve these problems, Ekberg and Yi[10] proposed a novel task model that calls the MS-DRT, which integrates arrival modes of jobs with global mode switching, and adopts a structured EDF-schedulability analysis approach where each mode of the system is relatively separated by abstracting the effects from other modes.

However, in FP scheduling, unbounded priority inversion can be prevented by using priority inheritance protocols,[69] but these protocols cannot work when zero-slack (ZS) schedulers are applied to schedule MC task-sets. Lakshmanan *et al.*[11] presented two protocols that extend real-time synchronization protocols like PCP[70] to harmonize the mode changes of the ZSs and to limit the criticality and priority inversions owing to resource sharing. They also developed methods to adjust the blocking terms caused by synchronization, for computing the ZS instants applied by the scheduler. Niz *et al.*[12] proposed an end-to-end ZS rate-monotonic method (ZSRM)[69] according to real-time pipelines, which is an MC scheduler developed for a

nonpipeline system. They calculated the ZS instant for each task in the task-set, and the resulting ZS instants make this task-set schedulable. Under ZSRM, each task is related with a parameter named ZS instant, and whenever an HC task is not completed at its ZS instant, all LC tasks are suspended to satisfy the deadline of the HC task. Niz and Phan[13] also proposed a partitioned scheduling method for multimodal MC-RTS on multiprocessor platforms. The scheduling approach is used for scheduling tasks on each processor. It extends the algorithm[69] with a mode transformation enforcement mechanism, which depends on the transitional ZS times during the mode change of the task to manage the LC tasks for keeping the schedulability of HC tasks. Execution time of LC tasks will be managed during mode changes for keeping the schedulability of HC tasks.

According to response time calculation,[71] the schedulability analysis approaches for systems with AMC scheduling are too complicated for optimization purposes. Zhao and Zeng[14] presented a schedulability analysis method according to request bound function (RBF)[72] for AMC-scheduled MC systems called AMC-RBF. To illustrate the motivation for developing AMC-RBF, they offered the formulation of the schedulable region according to AMC-RBF. Zhao *et al.*[15] proposed a protocol named HLC-PCP, which extends the PCP[73] for defending shared resources in AMC. They also presented a approach for worst case blocking time calculation with it, utilized for AMC schedulability analysis with shared resources, for both the general multicriticality model and the DC model. As locking time analysis in HLC-PCP only lies on the character of AMC scheduling algorithm, not on the algorithm of worst case response times (WCRT) analysis, they modified the WCRT analysis equations for AMC-RBF, which is used for computing WCRT value of each task for AMC,[65] to consider resource sharing by adding the terms of blocking time simply.

The widely used mode-switch methods supposed that all HC tasks are schedulable even the LC tasks are abandoned or degraded. However, this method activates a mode-switch at once after any task overruns, this can be pessimistic and abrupt. Considering this, Hu *et al.*[16] firstly solved MC systems scheduled, and presented light-weight mode-switch methods that can effectively move the system out of the critical mode. Its main idea is to execute overrun budget for all tasks, by supervising execution of the task and updating an ordinary overrun budget. To this end, the overrun budget can be adaptively complemented leveraging run-time information and shared among all tasks, thus, mode-switch can be delayed as much as possible. Experimental results showed that the proposed mode-switch reduces the mode-switch frequencies and abandoned jobs, as well as improves the scheduling time ratio of all tasks in the system.

However, the schedulability conditions from the above methods did not include the effect of system overheads, therefore, Chisholm *et al.*[17] considered an overhead called cache-related preemption delays (CRPDs), which are the delays that a task reloads lines out of the shared last-level caches (LLCs) according to a preemption. They utilized preemption-centric accounting to calculate CRPDs cost, where the

preempting job execution time is replenished to "pay" for the CPRD overheads that are caused by the execution resumption of each preemptive job when the preempting job is accomplished. Then, they formulated the schedulability conditions that accounted CRPD overheads for integrating these delays into schedulability analysis. Results showed that the proposed techniques can achieve schedulability improvements.

3. QoS-Oriented Techniques in MC Systems

However, these methods above guaranteed the timely executions of HC tasks at the expense of discarding LC tasks, which may cause serious service interruption for the tasks. In order to ensure schedulability of HC tasks, LC tasks are discarded immediately upon a critical change from LC to HC mode. This method completely ignores the LC tasks QoS. However, in some applications, it is still expectable to provide LC tasks with some (possibly degraded) QoS levels even in HC mode. In addition, it has been shown recently that task terminating could actually violate the security of the system.[45] We will introduce some solutions to these problems as follows.

According to a controlled randomization, Masrur[18] hence proposed a probabilistic scheduling framework for giving probabilistic real-time assurances on LC tasks, while providing deterministic timing assurances for HC tasks. In their model, jobs are not dropped to increase the schedulability of other tasks, but all tasks need to be verified to be within certain thresholds of failure probabilities depending on their criticalities and the criticality of the system. These tasks are scheduled under an FP scheme on one processor, for which priorities are allocated according to some policy.[63] Based on an FP preemptive scheduling policy, Abdeddam and Maxim[19] introduced a probabilistic MC model for MCRTSs operating on a single processor, where each task is of a given criticality and the WCET of the task is a random discrete variable. The set of possible execution times for each task are grouped into WCET sets of different criticalities according to their probability of occurrence. They also proposed an analysis to calculate response time assignments for tasks based on the Probabilistic MC. Besides these response time distributions, they can also extract deadline miss probabilities.

Al-bayati *et al.*[20] considered an scheduling model of MC system[74] that attempts to supply a decreased level of service for LC tasks, rather than simply discarding them.[65] They realized this goal by proposing a novel MC partitioning algorithm, called the dual-partitioned MC algorithm, which permits limited migration of LC tasks to improve the partitioning efficiency while keeping advantages of partitioned systems. This algorithm consists of two phases: optimization phase and partitioning phase. Huang *et al.*[21] extended the EDF-VD[62] scheduling method to ensure a degraded service to the LC tasks when the HC tasks exceed their LC WCET. They gave the approximation of demand bounds of all tasks under different operating modes, and defined the demand bound of a task in a given interval to be the sum of

the runtimes for all task instances. They also presented an analytical approach to bound offline the services resetting time provided to the LC tasks, and reconfigured the system with a simple runtime mechanism to ensure maximum degraded services for LC tasks. Zhao and Al-Bayati[22] used the Elastic MC task model for FP scheduling to supply a decreased level of QoS for LC tasks in High mode, and proposed a schedulability analysis method for elastic MC. As the optimization goal is to improve system services performance while ensuring schedulability, they proposed two optimization algorithms based on OPA[75] for allocating task priorities, and optionally adding delays to edges for producing a schedulable semantics-preserving realization.

However, degrading or terminating the services of less critical tasks may lead to great service/performance loss. To solve this problem, Huang *et al.*[23] used DVFS to accelerate the processor for guaranteeing that all tasks can still satisfy their deadlines in the case of timing urgency resulting in task overrun. They computed offline a minimum processor acceleration to ensure the schedulability of the system in critical operation mode. To ensure schedulability in Low mode, the sum of all tasks requiring bound functions in any time interval cannot be greater than the offered processing resource.[76] To provide ensured services for LC tasks, Su *et al.*[24] studied an elastic MC task model for DC systems. Specifically, the model permits each LC task to designate a set of early-release (ER) points and its minimum service level. After the LC task finishes executing the current instance, it can publish its next job instance at one of its ER points. To accommodate at runtime the ER executions of LC tasks, they proposed an ER mechanism that releases LC tasks more frequently and therefore increases their levels of service at run-time, while considering both aggressive and conservative methods for exploiting system slack. Liu *et al.*[25] researched real-time scheduling of MC system where LC tasks are still ensured some service in the High mode, while reducing execution budgets. They considered the imprecise MC model which increased the schedulability of LC tasks in High mode by decreasing their execution time. Since the task model is in Low mode, these tasks are considered as real-time tasks scheduled with the EDF algorithm that has a virtual deadline. And they proposed a schedulability test based on utilization of such systems using EDF-VD scheduling.[62] Gu and Easwaran[26] proposed a dynamic scheduling model of MC systems under which LC budgets for individual HC applications are decided at runtime rather than being fixed offline. To guarantee adequate budget for HC applications at all times, they used offline schedulability analysis for all the HC applications combined to decide a system-wide total LC budget allocation taking into account system-wide goals such as service guarantee and schedulability for LC applications. Then, the runtime policy allocates this budget to each application according to their execution need and delays mode switching as much as possible.

Hassan and Patel[27] introduced CArb, a configurable arbiter of criticality and requirement awareness for dominating accesses to shared memories and buses in multicore MC systems. CArb employs two-tier WRR[77] arbitration to manage these accesses. CArb optimally assigns service to tasks at startup through configurable

schedules loaded, and if the present set of memory needs for all tasks is non-schedulable, it gives priorities to tasks with higher criticality. CArb does not impose any restrictions on mapping tasks to processors and stands by arbitrary number of criticality levels. Hassan *et al.*[29] proposed a new method PMC for scheduling memory requests in MC systems. This method supports any number of criticality levels via allowing the MC system designer to appoint memory needs per task. They introduced a framework that builds optimal schedules to administer requests for off-chip memory and a tight time-division-multiplexing scheduler, and they employed a mixed-page strategy to dynamically switch between open-and close page strategies according to the size of the request. Guo and Pellizzoni[28] designed a new DRAM controller that executes and bundles memory requests of hard real-time (HRT) applications in consecutive rounds according to their type for significantly reducing read/write switching delay. Open-page is applied for soft real-time (SRT) requestors to increase bandwidth by using row buffer locality and Close-page scheme is applied for HRT requestors to realize predictability. Then, they proposed a request reordering scheduling policy to target systems with MC. The request scheduler executes different arbitration polices for SRT and HRT banks.

Kim *et al.*[30] solved shared-hardware interference caused by the operating system (OS). They presented a new platform, which extends a framework named MC2[78] by adding support for a variety of hardware management methods.[79] Specifically, they offered administration for both the DRAM[80] memory banks and LLC. To assign DRAM banks and LLC colors correctly to tasks, they built page pools for each bank and color combination. Resources that are shared among cores such as main memories, memory buses, and caches in their scheduling or analysis may also lead to much pessimism underlying the problem.[81] Chisholm *et al.*[31] firstly considered tradeoffs on multicore platforms for data sharing, where capacity loss is decreased by using both hardware-management methods[82] and MC configuration assumptions, and described a novel implementation of MC2, which extends the previous one by enabling tasks to communicate over shared memory, and offers approaches for administrating the DRAM memory banks and LLC. However, these approaches do not include inter-task interferences resulting from accessing resources which are shared among cores such as main memories, memory buses, and caches in their scheduling or analysis. Brandenburg[32] proposed a solution according to resource servers and presented MC-IPC, a novel synchronous multiprocessor inter-process communication (IPC) protocol for invoking such servers. The MC-IPC protocol allows strict logical and time isolation among untrusted tasks and therefore can be employed to share resources among different MC tasks.

Modern distributed interconnects either cannot meet the diverse needs or have decoupled arbitration phases, leading to larger area, power and so on. Gomony *et al.*[33] solved this problem by presenting globally arbitrated memory tree (GAMT). GAMT employs a distributed architecture that is configured with five arbitration strategies (FBSP,[83] CCSP,[83] TDM,[83] PBS[84] and RR[85]). They introduced the new

concept of mixed arbitration strategies, where the selection of arbiter is done by each client instead of being done jointly by all clients, to improve further arbitration flexibility without effecting cost. Then they proposed a new mixed arbitration strategy aiming at mixed-time-criticality systems. These mixed-time-criticality systems join nonwork-conserving TDM to realize time isolation with work-conserving FBSP to solve diversity and decrease average latency. Finally, they performed a worst-case analysis of the mixed strategy in the background of the LR[86] framework. Results showed that GAMT runs faster and saves power and area consumption, respectively.

4. Energy-Efficient Techniques in MC Systems

In the past two decades, energy management has become a key design and operational focus for many real-time embedded platforms. In fact, effective energy management is crucial to any battery-powered embedded system. In these systems, it is not always practical or feasible to recharge or replace the battery. Therefore, minimizing the energy consumption of embedded devices for a longer life expectancy has important practical and economic benefits. However, the existing approaches ensure the QoS of the system, which may lead to more energy consumption so that the MC systems cannot function normally or even collapse, which is a very serious threat for energy-limited MC systems. Therefore, studying energy optimization problem of the MC systems becomes inevitable. In this section, we will introduce some researches on optimizing the energy consumption of the MC system.

As the task's work may not be successfully implemented due to the lack of available energy, Asyaban *et al.*[34] considered MC systems with energy harvesters. These systems not only need to implement MC requirements within its deadline, but also need to meet its energy-related constraints. To ensure these constraints, at different moments, suitable task scheduling is required to manage the available energy of the super-capacitor, while taking into account the random pattern of energy arrival, the power consumption of the super-capacitor, and the energy usages of MC task. Therefore, they proposed a method according to the interaction of energy availability with MC scheduling and its analytical evaluation. Then, based on this analysis approach, they presented a scheduling algorithm which ensures the lower borders of job success probabilities. This algorithm is based on the features and nontrivial insights of RTSs, with a success rate constraint in the case of energy harvesting. The proposed algorithm is able to be further extended to deal with non-MC systems as well as MC systems with advanced loss models.

To guarantee timely completion of all tasks, while obeying this constraint, to do the best to conserve battery power, Awan *et al.*[35] mapped the given tasks on a heterogeneous multicore platform of different criticality levels by using partitioned scheduling such that available calculative power of the hardware platform can be exploited efficiently and energy cost is minimized while guaranteeing the timeliness

nature of the system. To realize this problem, they proposed an algorithm that optimizes the energy in the Low mode by manipulating the ordering of the tasks while respecting temporality. In this algorithm, initially, they rank the task-set for the computed density difference according to the energy cost in the Low mode and executed the distribution with the ILLED algorithm,[87] which ranks the tasks according to a metric named density difference while executing distribution to their preferred core. If the system is scheduled according to the rankings, they will achieve the minimum energy consumption allocation. Otherwise, they will arrange the task-set for computed density difference according to utilization in the Low mode and execute the distribution. They gradually improved the HC tasks in the ranking for getting a set of practical distributions and choosing the lowest energy consumption.

Volp *et al.*[36] re-evaluated the function of the energy budget in MC systems. Knowing that, in all FP scheduling methods those following the OCBP[88] produce the smallest acceleration bound.[89] They turned OCBP into an MC scheduler for energy constrained systems, by rearranging the sequence in which jobs are introduced to the OCBP assignment approach. They computed the maximum energy cost for all probable criticality level transforms of the tasks that are generated when scheduled with the proposed method, then they compared the results with the obtained schedules from an unsorted job list presented to OCBP. The results showed that compared to the unsorted OCBP scheduler, the proposed algorithm needs smaller budget, while still ensuring schedulability.

With drastically increased calculated needs and the battery-operated property of MC systems, the energy minimization of such systems is becoming crucial as well. Huang *et al.*[37] applied DVFS[90] to MC systems for minimizing the energy. The main idea of using DVFS to minimize energy cost is to stretch execution times of task as much as possible by reducing the processor frequency, so that tasks can be finished in time. They showed that DVFS is used to support critical tasks when they overrun to satisfy deadlines by accelerating the processor, which will further enable the system to maintain fewer time budgets. As overrun is few, such a scheme can greatly decrease the expected energy cost for MC systems. To achieve the energy minimization problem, they developed a convex program by combining DVFS with an MC scheduling method EDF-VD[63] that addresses DC tasks with implicit deadlines. Experimental results showed that the proposed methods are validated.

Narayana *et al.*[38] studied a general energy minimization issue for MC systems on multicores. They used a general setting in which both dynamic and static energy consumption are considered for all system operation modes. When making global scheduling policies, trade-offs in energy cost among different modes as well as between dynamic and static energy consumption are required. To solve the problem of energy cost in different modes and to minimize the total energy together, they divided the problem into two sub-problems: (1) first execute energy-aware task mapping; (2) then apply and develop unicore DVFS[90] method to all cores. As there is a convex formulation, they used the KKT optimality conditions[91] to solve this

problem. For partitioned scheduling, they proposed a new approach to realize this energy minimization problem by separating different criticality levels tasks on different cores. The results showed the proposed methods are valid, and can save energy.

Despite intensive studies on energy efficiency, these power-saving methods have not been applied in safety-critical areas. Lenz *et al.*[39] studied an architecture's requirements for a low-power MC system such as the fault tolerance and safety arguments, the power and energy efficiency, predictability and real-time features. Based on this, they introduced European project SAFE-POWER used to solve the problem of power management in MC systems real-time response, predictability, as well as power and energy efficiency requirements. SAFEPOWER constructs a set of comprehensive verification tools, simulation, and analysis for low-power MC systems, including software/hardware reference platforms assisting in observing, implementing and testing such applications. In the SAFEPOWER project, the generator of network-on-chip (NoC) system will be expanded to stand by NoC technologies with low-power and predictability. In addition, integrating Design space exploration (DSE) tools into NoC-system builders will benefit system design, as designers can then focus on application design while NoC-system builders generate full FPGA implementations and DSE tools compute efficient implementations.

Public buildings consume a significant percentage of energy in industrialized countries, thus building management systems must solve the problem of decreasing energy use. To realize this challenge, Kamienski[40] developed an internet of things (IoT) systems development platform (SDP) within the IMPReSS project, which aims at managing device's real-time prioritization for governing energy usage. IMPReSS SDP takes public buildings Energy Efficiency management as its first goal, but it is also available for any system aimed to embrace a more intelligent society. The platform contains a variety of components that make it easier to develop IoT applications, including IoT wireless communications management, analytics and management,[92] context-aware data storage, MC resource management,[93] pre-prepared middleware and software components, and tools to quickly develop user interfaces. In addition, typical middleware components for handling public buildings energy efficiency management are provided as well for hiding the implementation complexity from the application developers. Results showed that the IMPReSS SDP is effective for developing IoT-based and MC applications.

Different with the work above that reliability constraint is not taken into consideration, Li *et al.*[41] addressed the problem about how to schedule MC task sets to minimize energy consumption of the systems while also satisfying both deadline and reliability constraints. They first established the theoretical foundation for deciding whether the reliability and deadline constraints of the tasks can be meet with EDF-VD[62] scheduling method under given task execution frequency and virtual deadline allocations. According to the theoretic analysis, they developed a frequency assignment based on heuristic search algorithm that determines the execution frequency of

lowest task and minimizes energy consumption of the systems while ensuring both task schedulability and reliability constraints. Results showed that the proposed method can save more energy.

5. Fault-Tolerant Techniques in MC Systems

The above research has made tremendous efforts to develop techniques for dealing with the uncertainty in task schedulability, QoS and energy consumption of MC systems. MC system is susceptible to transient faults like other electronic systems. These systems must mitigate the effects of faults and provide recovery mechanisms when faults occur. However, few works take fault-tolerance as a design requirement, which represents the system repair capacity in the presence of faults. Therefore, researching fault tolerance makes sense for MC systems to enhance their robustness. We will present the research status in this section.

The schedule to meet real-time constraints can affect or be affected by the fault-tolerant constraints. Pathan[42] proposed a new method to model MC systems from the fault tolerant perspective. Then, they designed a uniprocessor FP scheduling algorithm, named fault-tolerant MC (FTMC) scheduling for this new model. To ensure both functional and temporal correctness, The FTMC algorithm performs backups to resume from task errors resulting from software or hardware faults. The task is considered to be erroneous, when a fault adversely influences the task functionality. Whenever a task's job is ready for execution, the FTMC algorithm first schedules the primary task. If primary task is found wrong, the backups are dispatched one by one until the output is proper. The backup has the same priority as the primary task. Results showed that the proposed test is effective.

Zhou *et al.*[43] solved the fault-tolerant MC tasks scheduling to guarantee the security of different levels of criticality tasks in the existence of transient faults on a uniprocessor platform. They proposed a fault-tolerant scheduling method called Slice-EDF-VD, which is based on EDF-VD.[62] Slice-EDF-VD algorithm uses re-execution to increase the reliability and adopts period transformation and utilization of idle time to improve schedule feasibility. With these two efforts, system safety can be upgraded. Results showed that Slice-EDF-VD can improve system schedule feasibility and reliability compared to existing approaches, helping to create safer systems.

Approaches such as module redundancy and rollback can effectively correct transient errors. However, in hard RTSs, the correction has a strong effect on the response time as well as tasks not directly influenced by the error. Here, the goal of Axer *et al.*[44] is to maximize reliability as much as possible to minimize oversupply. In order to solve this goal, they proposed a new algorithm. Unlike other approaches, they considered a representative hyperperiod, rather than the worst case, as it guarantees tighter reliability. The first step is to list all the feasible schemes for each job for the task over the entire hyperperiod. The second step is to turn these schemes

into probabilities by which deriving the feature reliability function is possible. The proposed algorithm showed a very good accuracy and significantly reduced analysis time for actual parameters.

Huang *et al.*[45] studied fault-tolerant MC scheduling problem under hardware transient faults. Based on established safety standards, they explicitly model the safety needs on different criticality levels. They analyzed the effects of service degradation, task killing, and task re-execution on system schedulability and safety. Then, they proposed a scheduling algorithm for this problem, in which all tasks of equal importance own the same re-execution profile. Furthermore, all HC tasks adaptation profiles are the same. The algorithm can be transformed into a conventional MC scheduling problem. Therefore, a large class of existing MC scheduling methods can be used. Extensive simulations showed that the proposed techniques are valid.

Zeng *et al.*[46] studied fault-tolerant MC scheduling under hardware/software transient faults on multicores. They proposed technologies that make it possible to reconfigure systems at runtime so that MC tasks can still be ensured under an emergency. They explicitly modeled safety needs on different criticality levels based on safety standards.[94] According to this, they further presented fault-tolerant MC scheduling methods with task re-execution and replication, and formulated the problem with adaptation and redundancy profiles to realize a feasible design. To cope with unsuccessful critical tasks under runtime emergencies, these methods can execute system reconfiguration to redistribute resources of the system to critical tasks. Because of explicit modeling of security, the effect of service degradation and task killing on system feasibility can be quantified, allowing a rigorous design. To this end, they distinguished between global and local reconstruction in these proposed techniques, enabling trade-offs between system feasibility and complexity analysis.

Bolchini and Miele[47] proposed a method and framework to implement embedded systems with MC fault management needs, according to an improved system-level synthesis. The proposed framework is a distributed one, consisting of resources with different performance characteristics and available fault tolerant mechanisms. The proposed method extends the traditional hardware/software coordination paradigm to stand by the standard of reliability-related needs, the application of fault tolerance methods, and the utilization of the architecture characteristics to realize a final system capable to control the transient failures. In this method, scheduling and mapping step is extended to stand by the application of hardening methods to achieve the required fault administration attributes that the final system has to display. Moreover, the method enables the designer to designate that only some sections of the systems require to be hardened to prevent faults.

Kang *et al.*[48] proposed an optimization method for fault-tolerant MC MPSoCs. Besides the traditional hardening methods by replication and re-execution, they also proposed a MC scheduling with dropped task that proves that HC applications offer

their service, and their WCRTs are ensured. The uncertainties caused by system hardening and MC algorithms, like dynamic task reduction, make it difficult to analyze the WCRTs for such systems. To tackle this challenge, they proposed a worst case analysis framework that considers over both reliability and MC concerns. Moreover, they constructed a space exploration engine to optimize fault-tolerant MC MPSoCs and offer worst case guarantees. Experiments confirmed that the proposed technique is effective.

Kang *et al.*[49] also proposed an optimization method of reliability-aware mapping for multicore MC systems, which is consistent with the existing standards. This technique is a universal framework that considers over judicious voter placement, active/passive replication, and its combinations, as well as re-execution. To allow a quantitative and comparative evaluation of the optimal mapping, they proposed a reliability metric based on probability. In order to cover the hardening method more comprehensively, they adopt selective voter arrangements and passive replication regarding the given limits. The fault-management method selected for each task to improve the reliability may lead to uncertain behaviors. To solve this problem, they used an analysis technique[95] that can be replaced by any existing method if it supports the variable execution time. Experimental results proved that the proposed approach is effective.

West *et al.*[50] introduced Quest-V,[96] which uses hardware virtualization to detach system components into sandboxes. Sandboxes administer their own subsets of performing scheduling and machine resources, I/O management, and memory without the participation of a hypervisor. Inter-sandbox communication is achieved by shared memory channels mapped to extended page tables (EPT) entries. Only trusted monitors can change entries in these EPTs to prevent visitors from accessing any memory areas in remote sandbox. It is probable to use N-versioning[97] or triple modular redundancy[98] methods to keep system operational, if a fault or security breach does happen in a monitor. Therefore, the design of Quest-V allows lower criticality services separated from higher criticality ones, and essential services replicated from different sandboxes to guarantee availability in the event of faults. Compared with traditional hypervisors, Quest-V system monitors own a small memory space, they are only used to divide resources at boot time, help fault recovery, and build inter-sandbox communication channels.

Al-bayati *et al.*[51] considered the issue of scheduling and designing certified fault-tolerant MC systems. To effectively deal with faults and task overruns in the system, they proposed a four-mode (transient faults mode, execution time overruns mode, and their combination mode) model that resolves execution time overrun and fault with separate modes when either transient faults or overruns occur. This model, integrated with the optional continuation of LC tasks, increases the QoS to these tasks while offering the same assurance to HC tasks. Experimental results showed that the proposed new model can improve reliability and schedule feasibility of the system while achieving LC task QoS improvements.

6. Specific Applications in MC Systems

With the rapid development of embedded systems, MC systems applications are more and more widely used. Typical applications include cyber-physical systems (CPSs), automotive systems,[99] grids and so on. We will introduce some main applications in MC systems below.

Petrakis *et al.*[52] introduced the integration of MPSoC with NoC in MC systems. They examined whether the NoC capacity is sufficient to meet deadlines for external HC workloads while also serving the lower internal critical workloads. In order to evaluate the NoC, they changed its configuration parameters such as QoS, link width, and relative priority of packets from different streams sharing the same virtual channel. Another traffic scenario is to assign priorities to different traffic sharing the same virtual channel. This is realized by QoS fair bandwidth distribution, a low-cost arbitration method that distributes network interface objective bandwidth during peak demand times among different initiators. The results showed that this method can balance the cost and performance.

NoC must offer performance segregation for security-critical traffic, while maintaining low latency for traffic. Tobuschat and Ernst[53] proposed a run-time configurable design of NoC that enables latency assurances for security-critical traffic with decreased adverse effect on the performance of traffic. They prioritize Guaranteed Delay traffic in NoC routers and switched priorities as needed based on actual congestion. Based on the actual blocking, they prioritized ensured-delay traffic in NoC routers and just switched priorities when needed. For this, they derived the slack time for the flow through timing analysis and saved the slack time in the packet header. The slack time is then evaluated and modified in the router to administer the leftover slack time. This allows using critical applications latency slack, while offering enough independence between different criticality levels regarding timing properties. Experimental results showed that the approach offers enough isolation, while decreasing the adverse influences for safety-critical applications.

Effective scheduling strategies can take full advantage of electronic control units in automotive CPS for high performance. However, automotive CPS should address the common challenges of parallelism, dynamism, heterogeneity, criticality and security. To solve these challenges, Xie *et al.*[54] first presented a dynamic scheduling algorithm based on fairness called FDS_MIMF to minimize schedule lengths from a high-performance angle. FDS_MIMF is able to respond autonomously to the common challenge of parallelism, dynamism and heterogeneity of ACPS. To further respond to these challenges, they proposed an adaptive dynamic scheduling called ADS_MIMF to realize low deadline miss rates of the functions from a timing constraint angle while keeping the overall acceptable automotive CPS's make-span from a high-performance point of view. ADS_MIMF is achieved by reducing and increasing the criticality level of automotive CPS in order to adjust the implement of different functions at different criticality levels without improving the time

complexity. Experiments showed that FDS_MIMF is able to get shorter overall make-span, ADS_MIMF can decrease the value of deadline miss rates of HC functions while maintaining high performance of Automotive CPS.

Smirnov *et al.*[55] considered a MCS from the automotive domain and introduced Pseudo-Boolean constraints which guarantees scheduled messages can have a valid global schedule and a valid routing in the network. These constraints can be used to produce an effective MC communication network in a single step that has scheduled traffic. Then, they extended these constraints for combining it with the constraints[100] and using it to generate a valid schedule, which in a single step allocates the required global offset to all scheduled messages. They used a similar idea,[101] where the schedule is produced by developing SMT-constraints[102] for two possible pairs of overlapping transmission slots. But they reformulated them as Pseudo-Boolean constraints. Experimental results showed that the approach performs better for hard scheduling problems, and is suitable for the MC systems multiobjective optimization.

Zhao *et al.*[56] solved the design comprehensive problem of executing the AUTOSAR model in automotive MC systems, where each task is composed by multiple operational plans with preemption thresholds and static priority. For the priority assignment and mappings from operational to tasks, they used the PA-DMMPT algorithm[103] to set task priorities. By allocating higher priorities to higher criticality tasks, they modified PA-DMMPT to break a tie when utilizing Deadline Monotonic. Then they presented a heuristic algorithm named HeuPADMMPT which extends PA-DMMPT by limiting that only the same critical runnable are able to be mapped to the same task. Results showed that HeuPADMMPT can significantly decrease the usage of the system stack.

Recently, Cluster-based Scheduling has become increasingly important for using real-time MC systems on multicore processor platforms. In these approaches, the cores are divided into clusters, and each cluster of the global scheduler schedules the partitioned tasks among different clusters. Ali and Kim[30] introduced a novel cluster-based task distribution method for the real-time task sets on multicore processors in MC systems. For task allocation, smaller sizes of clusters are utilized for MC tasks under LC mode, relatively larger sizes of cluster are utilized for HC tasks under High mode. Here the set of MC task is assigned to clusters employing worst-suit heuristic. Tasks from each cluster are assigned to its sub-clusters too, using the same worst-suit heuristic. For schedulability analysis, they used FP RTA[104] according to Audsley's approach for cluster of the set of MC task and each sub-cluster. Results showed that the proportion of schedulable task sets under cluster scheduling significantly improves compared with global and partitioned MC scheduling methods.

Giannopoulou *et al.*[58] extended the state-of-the-art for MC systems by presenting a unified analysis approach for computing, memory, and communication scheduling. To model such communication flows and architectures through the NoC, they concretized and extended the system model introduced in work.[105] In addition, they introduced an inter-cluster communication protocol with formally proven timing features. To schedule MC applications on an architecture based on cluster, they

proposed an MC scheduling method called FTTS that implements global timing isolation between different critical applications to provide demonstrable features. This is realized by enabling only applications of equal importance to be implemented in parallel and therefore, disturb the communication infrastructure and shared memory. The results showed that FTTS performs better in schedulability.

Standard Ethernet cannot supply hard latency assurances needed for distributed safety-critical applications like industrial control, automobiles, and avionics.[106] Carvajal *et al.*[59] introduced a real-time ethernet framework named Atacama for multisegmented MC traffic networks. Atacama employs a time-triggered method,[106] which integrates an ASIP[107] to coordinate the switch of time-sensitive data for each station performing real-time tasks. Moreover, Atacama adopts a custom forwarding path that supplies predictable and low propagation delay across multiple switches. With the experimental data that implements the prototype, they derived a communication delay model of real-time frames. The model offers a precise upper-limit on the end-to-end delay among distributed real-time tasks, limited only by physical characteristics like the drift between uncertainty and clock domains in the physical links. Experiments showed that latency assurances are robust for the best-effort traffic. Atacama enables researcher to validate the devices, build upon, and test the resources on their applications.

Compared to common deadline-driven scheduling methods, scheduling MCCPS needs a combination of QoC-driven and timing-driven methods, which causes a challenging scheduling problem. To tackle this problem, Schneider *et al.*[60] proposed a multilayered scheduling algorithm for MC CPS composed of feedback control tasks and HRT tasks. The proposed algorithm integrates timing analysis methods based on RTC[108] to an MLS framework. Real-time tasks can be scheduled in the top layer based on a timing-driven scheduling scheme, and control tasks are allocated priorities in the second layer subject to QoC optimization. The proposed approach significantly increases overall QoC while ensuring schedulability.

New challenges for the partitioned systems include the usage of distribution standards and multiprocessor architectures for opening up this method to CPSs. To solve these challenges, Prez *et al.*[61] studied three policies: (1) the usage of a multiprocessor method that enables one core to be specially allocated for communications, which avoids the inherent additional delays to the time window configuration. (2) The usage of priority-based scheduling for deciding the order in which partitions are executed at the partition level. (3) The combination of scheduling schemes in a multiprocessor method that enables space and time isolation to be ensured in a set of cores, therefore it may perform partitions with authentication requirements.

7. Conclusions and Future Directions

In this paper, we introduce the research status of the schedulability in MC systems. Then, we investigate some techniques to solve the problem of QoS caused by

improving schedulability. However, as improving the QoS may inevitably lead to excessive energy consumption, we introduce the technologies for minimizing the energy consumption of MC systems. Fault-tolerance is ignored in most MC systems, so we investigate the researches that take it as a design requirement. Finally, we introduce several key applications in MC systems. We believe that in the future MC systems will attract more attention and their applications will be used more widely. It will be helpful to study more design requirements of MC systems for making MC systems more secure.

Acknowledgments

The authors are very grateful to the editor and reviewers for their suggestions to improve the quality of paper. This work was supported by the National Natural Science Foundation of China (61802185, 61872185, 61502234, and 61272420), and the Natural Science Foundation of Jiangsu Province of China (BK20180470 and BK20150785), and the Science & Technology Support Project of Jiangxi Province (GJJ150424).

References

1. S. Baruah, B. Chattopadhyay, H. Li and I. Shin, Mixed-criticality scheduling on multiprocessors, *Real-Time Syst.*, Vol. 50 (Berlin, Germany, 2014), pp. 142–177.
2. L. Sigrist, G. Giannopoulou, P. Huang, A. Gomez and L. Thiele, Mixed-criticality runtime mechanisms and evaluation on multicores, *IEEE Real-Time and Embedded Technology and Applications Symp.* (Seattle, WA, USA, 2015), pp. 194–206.
3. S. Baruah, Schedulability analysis for a general model of mixed-criticality recurrent real-time tasks, *Real-Time Systems Symp.* (Porto, Portugal, 2017), pp. 25–34.
4. S. Ramanathan and A. Easwaran, Utilization difference based partitioned scheduling of mixed-criticality systems, *Design, Automation & Test in Europe Conference & Exhibition (DATE)* (Lausanne, Switzerland, 2017), pp. 238–243.
5. J. J. Han, X. Tao, D. Zhu and H. Aydin, Criticality-aware partitioning for multicore mixed-criticality systems, *Int. Conf. Parallel Processing (ICPP)* (Philadelphia, PA, USA, 2016), pp. 227–235.
6. E. Yip, M. M. Y. Kuo, P. S. Roop and D. Broman, Relaxing the synchronous approach for mixed-criticality systems, *Real-Time and Embedded Technology and Applications Symposium (RTAS)* (Berlin, Germany, 2014), pp. 89–100.
7. N. Guan, P. Ekberg, M. Stigge and W. Yiz, Effective and efficient scheduling of certifiable mixed-criticality sporadic task systems, *IEEE 32nd Real-Time Systems Symp.* (Vienna, Austria, 2011), pp. 13–23.
8. Y. Chen, G. S. Kang and H. Xiong, Generalizing fixed-priority scheduling for better schedulability in mixed-criticality systems, *Inf. Process. Lett.* **116** (2016) 508–512.
9. B. Hu, K. Huang, G. Chen, L. Cheng, D. Han and A. Knoll, Schedulability analysis towards arbitrarily activated tasks in mixed-criticality systems, *J. Circuits Syst. Comput.* **26** (2017) 1750159.
10. P. Ekberg and Y. Wang, Schedulability analysis of a graph-based task model for mixed-criticality systems, *Real-Time Syst.*, Vol. 52 (Kluwer Academic Publishers Norwell, MA, USA, 2016), pp. 1–37.

11. K. Lakshmanan, D. D. Niz and R. Rajkumar, Mixed-criticality task synchronization in zero-slack scheduling, *IEEE Real-Time and Embedded Technology and Applications Symp.* (Austin, TX, USA, 2011), pp. 47–56.
12. D. de Niz, B. Andersson, H. Kim, M. Klein, L. T. X. Phan and R. Rajkumar, Mixed-criticality processing pipelines, *Design, Automation & Test in Europe Conf. & Exhibition (DATE)* (Lausanne, Switzerland, 2017), pp. 1372–1375.
13. D. D. Niz and L. T. X. Phan, Partitioned scheduling of multi-modal mixed-criticality real-time systems on multiprocessor platforms, *Real-Time and Embedded Technology and Applications Symp. (RTAS)* (Berlin, Germany, 2014), pp. 111–122.
14. Y. Zhao and H. Zeng, An efficient schedulability analysis for optimizing systems with adaptive mixed-criticality scheduling, *Real-Time Syst.* **53** (2017) 1–59.
15. Q. Zhao, Z. Gu, M. Yao and H. Zeng, HLC-PCP: A resource synchronization protocol for certifiable mixed criticality scheduling, *J. Syst. Archit.* **6** (2014) 8–11.
16. B. Hu, K. Huang, P. C. Huang, L. Thiele and A. Knoll, On-the-fly fast overrun budgeting for mixed-criticality systems, *Int. Conf. Embedded Software (EMSOFT)* (Pittsburgh, PA, USA, 2016), pp. 1–10.
17. M. Chisholm, B. C. Ward, N. Kim and J. H. Anderson, Cache sharing and isolation tradeoffs in multicore mixed-criticality systems, *IEEE Real-Time Systems Symp.* (San Antonio, TX, USA, 2015), pp. 305–316.
18. A. Masrur, A probabilistic scheduling framework for mixed-criticality systems, *ACM/EDAC/IEEE Design Automation Conference (DAC)* (Austin, TX, USA, 2016), pp. 1–6.
19. Y. Abdeddam and D. Maxim, Probabilistic schedulability analysis for fixed priority mixed criticality real-time systems, *Automation & Test in Europe Conference & Exhibition (DATE)* (Lausanne, Switzerland, 2017), pp. 596–601.
20. Z. Al-bayati, Qingling Zhao, A. Youssef, H. Zeng and Z. Gu, Enhanced partitioned scheduling of mixed-criticality systems on multicore platforms, *Asia and South Pacific Design Automation Conf.* (Chiba, Japan, 2015), pp. 630–635.
21. P. Huang, G. Giannopoulou, N. Stoimenov and L. Thiele, Service adaptions for mixed-criticality systems, *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)* (Singapore, 2014), pp. 125–130.
22. Q. Zhao and Z. Al-Bayati, Optimized implementation of multirate mixed-criticality synchronous reactive models, *ACM Trans. Design Autom. Electron. Syst. (TODAES)*, Vol. 22 (New York, NY, USA, 2017), p. 23.
23. P. Huang, P. Kumar, G. Giannopoulou and L. Thiele, Run and be safe: Mixed-criticality scheduling with temporary processor speedup, *Design, Automation & Test in Europe Conference & Exhibition (DATE)* (Grenoble, France, 2015), pp. 1329–1334.
24. H. Su, D. Zhu and S. Brandt, An elastic mixed-criticality task model and early-release EDF scheduling algorithms, *ACM Trans. Design Autom. Electron. Syst.* **22** (2017) 1–25.
25. D. Liu *et al.*, EDF-VD scheduling of mixed-criticality systems with degraded quality guarantees, *IEEE Real-Time Systems Symp. (RTSS)* (2016), pp. 35–46.
26. X. Gu and A. Easwaran, Dynamic budget management with service guarantees for mixed-criticality systems, *IEEE Real-Time Systems Symp. (RTSS)* (Porto, Portugal, 2016), pp. 47–56.
27. M. Hassan and H. Patel, Criticality- and requirement-aware bus arbitration for multicore mixed criticality systems, *IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS)* (Vienna, Austria, 2016), pp. 1–11.
28. D. Guo and R. Pellizzoni, A requests bundling DRAM controller for mixed-criticality systems, *IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS)* (Pittsburgh, PA, USA, 2017), pp. 247–258.

29. M. Hassan, H. Patel and R. Pellizzoni, PMC: A requirement-aware DRAM controller for multi-core mixed criticality systems, *ACM Trans. Embed. Comput. Syst.*, Vol. 16 (New York, NY, USA, 2017), pp. 100–128.
30. N. Kim, B. C. Ward, M. Chisholm, C. Y. Fu, J. H. Anderson and F. D. Smith, Attacking the one-out-of-m multicore problem by combining hardware management with mixed-criticality provisioning, *IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS)* (Vienna, Austria, 2016), pp. 1–12.
31. M. Chisholm, N. Kim, B. C. Ward, N. Otterness, J. H. Anderson and F. D. Smith, Reconciling the tension between hardware isolation and data sharing in mixed-criticality (Porto, Portugal, 2016), pp. 57–68.
32. B. B. Brandenburg, A synchronous IPC protocol for predictable access to shared resources in mixed-criticality systems, *IEEE Real-Time Systems Symp.* (Rome, Italy, 2014), pp. 196–206.
33. M. D. Gomony, J. Garside, B. Akesson, N. Audsley and K. Goossens, A globally arbitrated memory tree for mixed-time-criticality systems, *IEEE Trans. Comput.* **66** (2017) 212–225.
34. S. Asyaban, M. Kargahi, L. Thiele and M. Mohaqeqi, Analysis and scheduling of a battery-less mixed-criticality system with energy uncertainty, *ACM Trans. Embed. Comput. Syst.*, Vol. 16 (New York, NY, USA, 2016), p. 23.
35. M. A. Awan, D. Masson and E. Tovar, Energy-aware task allocation onto unrelated heterogeneous multicore platform for mixed criticality systems, *IEEE Real-Time Systems Symp.* (San Antonio, TX, USA, 2015), pp. 377–377.
36. M. Volp, M. Hhnel and A. Lackorzynski, Has energy surpassed timeliness? Scheduling energy-constrained mixed-criticality systems, *IEEE 19th Real-Time and Embedded* (Berlin, Germany, 2014), pp. 275–284.
37. P. Huang, P. Kumar, G. Giannopoulou and L. Thiele, Energy efficient DVFS scheduling for mixed-criticality systems, *Int. Conf. Embed. Software (EMSOFT)* (Jaypee Greens, India, 2014), pp. 1–10.
38. S. Narayana, P. Huang, G. Giannopoulou, L. Thiele and R. V. Prasad, Exploring energy saving for mixed-criticality systems on multi-cores, *IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS)* (Vienna, Austria, 2016), pp. 1–12.
39. A. Lenz *et al.*, SAFEPOWER project: Architecture for safe and power-efficient mixed-criticality systems, *Euromicro Conf. Digital System Design (DSD)* (Limassol, Cyprus, 2016), pp. 294–300.
40. C. Kamienski, Application development for the internet of things: A context-aware mixed criticality systems development platform, *Computer Communications* **104** (2017) 1–16.
41. Z. Li, C. Guo, X. Hua and S. Ren, Reliability guaranteed energy minimization on mixed-criticality systems, **112** (2016) 1–10.
42. R. M. Pathan, Fault-tolerant and real-time scheduling for mixed-criticality systems, *Real-Time Syst.*, Vol. 50 (Kluwer Academic Publishers Norwell, MA, USA, 2014), pp. 509–547.
43. J. Zhou, M. Yin, Z. Li, K. Cao and J. M. Yan, Fault-tolerant task scheduling for mixed-criticality real-time systems, *J. Circuits Syst. Comput.* **26** (2017).
44. P. Axer, M. Sebastian and R. Ernst, Reliability analysis for MPSoCs with mixed-critical, hard real-time constraints, *Proc. Ninth IEEE/ACM/IFIP Int. Conf. Hardware/Software Codesign and System Synthesis (CODES+ISSS)* (Taipei, Taiwan, 2011), pp. 149–158.
45. P. Huang, H. Yang and L. Thiele, On the scheduling of fault-tolerant mixed-criticality systems, *ACM/EDAC/IEEE Design Automation Conf. (DAC)* (San Francisco, CA, USA, 2014), pp. 1–6.

46. L. Zeng, P. Huang and L. Thiele, Towards the design of fault-tolerant mixed-criticality systems on multicores, *Int. Conf. Compliers, Architectures, and Sythesis of Embedded Systems (CASES)* (Pittsburgh, PA, USA, 2016), pp. 1–10.
47. C. Bolchini and A. Miele, Reliability-driven system-level synthesis for mixed-critical embedded systems, *IEEE Trans. Comput.* **62** (2013) 2489–2502.
48. S. H. Kang, H. Yang, S. Kim, I. Bacivarov, S. Ha and L. Thiele, Static mapping of mixed-critical applications for fault-tolerant MPSoCs, *ACM/EDAC/IEEE Design Automation Conf. (DAC)* (San Francisco, CA, USA, 2014), pp. 1–6.
49. S. H. Kang, H. Yang, S. Kim, I. Bacivarov, S. Ha and L. Thiele, Reliability-aware mapping optimization of multi-core systems with mixed-criticality, *Design, Automation & Test in Europe Conf. & Exhibition (DATE)* (Dresden, Germany, 2014), pp. 1–4.
50. R. West, Y. Li, E. Missimer and M. Danish, A virtualized separation kernel for mixed-criticality systems, *ACM Trans. Comput. Syst.* **34** (2016) 201–212.
51. Z. Al-bayati, J. Caplan, B. H. Meyer and H. Zeng, A four-mode model for efficient fault-tolerant mixed-criticality systems, *Design, Automation & Test in Europe Conf. & Exhibition (DATE)* (Austin, TX, USA, 2016), pp. 97–102.
52. P. Petrakis *et al.*, On-chip networks for mixed-criticality systems, *IEEE 27th Int. Conf. Application-Specific Systems, Architectures and Processors (ASAP)* (London, UK, 2016), pp. 164–169.
53. S. Tobuschat and R. Ernst, Efficient latency guarantees for mixed-criticality networks-on-chip, *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)* (Pittsburgh, PA, USA, 2017), pp. 113–122.
54. G. Xie, G. Zeng, Z. Li, R. Li and K. Li, Adaptive dynamic scheduling on multifunctional mixed-criticality automotive cyber-physical systems, *IEEE Transactions on Vehicular Technol.* **66** (2017) 6676–6692.
55. F. Smirnov, M. Gla, F. Reimann and J. Teich, Optimizing message routing and scheduling in automotive mixed-criticality time-triggered networks, *ACM/EDAC/IEEE Design Automation Conference (DAC)* (Austin, TX, USA, 2017), pp. 1–6.
56. Q. Zhao, Z. Gu and H. Zeng, Design optimization for AUTOSAR models with preemption thresholds and mixed-criticality scheduling, *J. Syst. Archit.* **72** (2016) 61–68.
57. A. Ali and K. H. Kim, Cluster-based multicore real-time mixed-criticality scheduling, *J. Syst. Archit.* **79** (2017) 45–58.
58. G. Giannopoulou, N. Stoimenov, P. Huang and L. Thiele, Mixed-criticality scheduling on cluster-based manycores with shared communication and storage resources, *Real-Time Syst.* **52** (2016) 399–449.
59. G. Carvajal, M. Figueroa, R. Trausmuth and S. Fischmeister, Atacama: An open FPGA-based platform for mixed-criticality communication in multi-segmented ethernet networks, *IEEE 21st Annual Int. Symp. Field-Programmable Custom Computing Machines* (Seattle, WA, USA, 2013), pp. 121–128.
60. R. Schneider, D. Goswami, A. Masrur, M. Becker and S. Chakraborty, Multi-layered scheduling of mixed-criticality cyber-physical systems, *J. Syst. Archit. Euromicro J.* **59** (2013) 1215–1230.
61. H. Prez, J. J. Gutirrez, S. Peir and A. Crespo, Distributed architecture for developing mixed-criticality systems in multi-core platforms, *J. Syst. Softw.* **123** (2016) 145–159.
62. S. Baruah, Optimal utilization bounds for the fixed-priority scheduling of periodic task systems on identical multiprocessors, *IEEE Trans. Comput.* **53** (2004) 781–784.
63. S. Baruah, V. Bonifaci, G. DAngelo, H. Li, A. Marchetti-Spaccamela, S. van der Ster and L. Stougie, The preemptive uniprocessor scheduling of mixed-criticality implicit-deadline sporadic task systems, *Real-Time Syst.* **62** (2012) 145–154.

64. G. Giannopoulou, N. Stoimenov, P. Huang and L. Thiele, Scheduling of mixed-criticality applications on resource-sharing multicore systems, *Int. Conf. Embedded Software (EMSOFT)* (Montreal, QC, Canada, 2013), pp. 1–15.
65. S. K. Baruah, A. Burns and R. I. Davis, Response-time analysis for mixed criticality systems, *Real-Time Systems Symposium (RTSS)* (2011), pp. 34–43.
66. S. Baruah, V. Bonifaci, G. DAngelo, H. Li, A. Marchetti-Spaccamela, S. Van Der Ster and L. Stougie, Preemptive uniprocessor scheduling of mixed-criticality sporadic task systems, *J. ACM* **62** (2015) 1–33.
67. H. Su, D. Zhu and D. Mosse, Scheduling algorithms for elastic mixed-criticality tasks in multicore systems, *IEEE Int. Conf. Embedded and Real-Time Computing Systems and Applications (RTCSA)* (Taipei, Taiwan, 2013), pp. 352–357.
68. R. Henia, A. Hamann, M. Jersak and R. Racu, System level performance analysis — the SymTA/S approach, *IEE Proc.-Comput. Digital Tech.* **152** (2005) 148–166.
69. D. D. Niz, K. Lakshmanan and R. Rajkumar, On the scheduling of mixed-criticality real-time task sets, *Real-Time Syst. Symp.* (2009) 291–300.
70. L. Sha, R. Rajkumar and J. Lehoczky, Priority inheritance protocols: An approach to real-time synchronization, *Computers, IEEE Trans. Comput.* **39** (1990) 1175–1185.
71. S. K. Baruah, Dynamic- and static-priority scheduling of recurring real-time tasks, *Real-Time Syst.* **24** (2003) 93–128.
72. H. Zeng and M. D. Natale, An efficient formulation of the real-time feasibility region for design optimization, *IEEE Trans. Comput.* **62** (2013) 644–661.
73. J. B. Goodenough and L. Sha, The priority ceiling protocol: A method for minimizing the blocking of high priority Ada tasks, *ACM SIGAda Ada Lett.* **7** (1988) 20–31.
74. A. Burns and S. Baruah, Towards a more practical model for mixed criticality systems, *IEEE Real-Time Systems Symp.* (2013).
75. N. C. Audsley, On priority assignment in fixed priority scheduling, *Information Process. Lett.* **79** (2001) 39–44.
76. P. Ekberg and W. Yi, Bounding and shaping the demand of generalized mixed-criticality sporadic task systems, *Real-Time System* **50**(1) (2014) 48–86.
77. M. Katevenis, S. Sidiropoulos and C. Courcoubetis, Weighted round- robin cell multiplexing in a general-purpose atm switch chip, *IEEE Journal on Selected Areas in Commun.* **9** (1991) 1265–1279.
78. J. L. Herman, C. J. Kenna, M. S. Mollison, J. H. Anderson and D. M. Johnson, RTOS support for multicore mixed-criticality systems, *IEEE, Real Time Embed. Technol. Appl. Symp.* (Beijing, China, 2012), pp. 197–208.
79. J. Herman, C. Kenna, M. Mollison, J. Anderson and D. Johnson, RTOS support for multicore mixed-criticality systems, *IEEE Real-Time Embed. Technol. Appl. Symp.*, Vol 282 (Beijing, China, 2012), pp. 197–208.
80. L. Liu, Z. Cui, M. Xing, Y. Bao, M. Chen and C. Wu, A software memory partition approach for eliminating bank-level interference in multicore systems, *ICPACT* (Minneapolis, MN, USA, 2017), pp. 367–376.
81. J. Erickson, N. Kim and J. Anderson, Recovering from overload in multicore mixed-criticality systems, *IEEE Int. Parallel and Distributed Processing Symp.* (2015), pp. 775–785.
82. G. Xie, Y. Chen, R. Li and K. Li, Hardware cost design optimization for functional safety-critical parallel applications on heterogeneous distributed embedded systems, *IEEE Trans. Indust. Inform.* (2017) 1–1.

83. B. Akesson and K. Goossens, *Memory Controllers For Real-Time Embedded Systems* (Springer, New York, 2011).
84. M. Steine, M. Bekooij and M. Wiggers, A priority-based budget scheduler with conservative dataflow model, *Architectures Methods Tools* (Patras, Greece, 2009), pp. 37–44.
85. A. Rahimi, I. Loi, M. Kakoee and L. Benini, A fully-synthesizable single-cycle interconnection network for shared-L1 processor clusters, *Test Europe Conf. Exhibition* (Grenoble, France, 2011), pp. 1–6.
86. D. Stiliadis and A. Varma, Latency-rate servers: A general model for analysis of traffic scheduling algorithms, *IEEE Trans* **6** (1998) 611–624.
87. M. A. Awan, P. M. Yomsi, G. Nelissen and S. M. Petters, Energy-aware task mapping onto heterogeneous platforms using DVFS and sleep states, *Real-Time Syst.* **52** (2016) 450–485.
88. S. Baruah, V. Bonifaci, G. D. Angelo, H. Li, A. Marchetti-Spaccamela, N. Megow and L. Stougie, Scheduling real-time mixed-criticality jobs, *IEEE Trans. Comput.* **61** (2012).
89. C. Gu, N. Guan, Q. Deng and W. Yi, Improving OCBP-based scheduling for mixed-criticality sporadic task systems, *Int. Conf. Embedded and Real-Time Computing Systems and Applications* (IEEE, Taipei, Taiwan, 2013), pp. 247–256.
90. P. Huang, O. Moreira, K. Goossens and A. Molnos, Throughput-constrained voltage and frequency scaling for real-time heterogeneous multiprocessors, *Proc. 28th Annual ACM Symp. Applied Computing* (2013), pp. 1517–1524.
91. H. Kuhn and A. Tucker, Nonlinear programming, *Second Berkeley Symposium on Mathematical Statistics and Probability* (1951), pp. 481–492.
92. C. Perera, A. Zaslavsky, P. Christen and D. Georgakopoulos, Context aware computing for the Internet of Things: A survey, *IEEE Commun. Surveys Tutorials* **16** (2014) 414–454.
93. A. Burns and R. I. Davis, Mixed criticality systems: A review, Department of Computer Science University of York Tech. Rep. MCC-1(b) (2013).
94. S. Brown, Overview of IEC 61508. design of electrical/electronic/programmable electronic safety-related systems, *Comput. Control Eng. J.* **11** (2000) 6–12.
95. J. Kim *et al.*, A novel analytical method for worst case response time estimation of distributed embedded systems, *Proc. 50th Annual Design Autom. Conf.*, Vol. 129 (Austin, TX, USA, 2013), p. 10.
96. J. M. Rushby, Design and verification of secure systems, *Assoc. Comput. Mach.* **15** (1981) 12–21.
97. A. Avizienis, The n-version approach to fault-tolerant software, *IEEE Trans. Software Eng.* **11**(12) (1985) 1491–1501.
98. R. E. Lyons and W. Vanderkulk, The use of triple-modular redundancy to improve computer reliability, *IBM J. Res. Dev.* **6** (1962) 200–209.
99. G. Xie, G. Zeng, Y. Liu, J. Zhou, R. Li and K. Li, Fast functional safety verification for distributed automotive applications during early design phase, *IEEE Trans. Ind. Electron.* **65** (2018) 4378–4391.
100. M. Lukasiewycz, S. Shreejith and S. A. Fahmy, System simulation and optimization using reconfigurable hardware, *Int. Symp. Integrated Circuits (ISIC)* (Singapore, 2014), pp. 468–471.
101. F. Sagstetter, M. Lukasiewycz and S. Chakraborty, Schedule integration for time-triggered systems, *Design Automation Conf. (ASP-DAC), Asia and South Pacific* (Yokohama, Japan, 2013), pp. 53–58.
102. W. Steiner, An evaluation of smt-based schedule synthesis for time-triggered multi-hop networks, *Real-Time Systems Symposium (RTSS)* (San Diego, CA, USA, 2010), pp. 375–384.

103. H. Zeng, M. D. Natale and Q. Zhu, Minimizing stack and communication memory usage in real-time embedded applications, *ACM Trans. Embedded Comput. Syst.* **149** (2014) 1–149.
104. N. Audsley, A. Burns, M. Richardson, K. Tindell and A. Wellings, Applying new scheduling theory to static priority pre-emptive scheduling, *Softw. Eng. J.* **8** (2002) 284–292.
105. G. Giannopoulou, N. Stoimenov, P. Huang and L. Thiele, Scheduling of mixed-criticality applications on resource-sharing multicore systems, *EMSOFT* (Montreal, QC, Canada, 2013), pp. 1–15.
106. H. Kopetz, The rationale for time-triggered ethernet, *Real-Time Syst. Symp.* (Barcelona, Spain, 2008), pp. 3–11.
107. S. Fischmeister, R. Trausmuth and I. Lee, Hardware acceleration for conditional state-based communication scheduling on real-time ethernet, *IEEE Trans. Ind. Inf.* **5** (2009) 325–337.
108. S. Chakraborty, S. Knzli and L. Thiele, A general framework for analyzing system properties in platform-based embedded system designs, *Design, Automation & Test in Europe Conf. & Extibition (DATE)* (Munich, Germany, 2003).