Evolutionary Arbiter Based on Adaptive Weight Evolution Mechanism

Jianhao Huang

College of Optoelectronic and Information Engineering, Fujian Normal University, Fuzhou, 350100, China
136132023053@student.fjnu.edu.cn

Abstract. Traditional static arbiters cannot cope with scenarios with large fluctuations in resource demand, especially in multilateral bus systems and network switching. In recent years, evolutionary arbiter (EVOA) based on evolutionary algorithms provides a flexible and efficient resource scheduling mechanism in scenarios with dynamic loads and changing resource demands due to its flexible adaptive weight adjustment mechanism. This paper reviews recent research on evolutionary arbiters, focusing on the application of the adaptive weight evolution mechanism in optimizing task prioritization and resource allocation. The results show that the evolutionary arbiter can effectively cope with dynamic load changes, improve resource utilization and ensure system fairness. The evolutionary arbiter demonstrates higher bandwidth utilization and system throughput compared to the traditional static arbiter in multilateral bus systems and data centre environments. In addition, in network switching, evolutionary arbiters can effectively avoid resource contention and improve the efficiency and fairness of the network. Although evolutionary arbiters excel in performance optimization, their computational complexity and hardware implementation still face challenges. Future research will focus on algorithm optimization, computational efficiency improvement and hardware implementation to further enhance the application scope and practicality of evolutionary arbiters.

Keywords: Evolutionary Arbiter, Adaptive Weights, Evolutionary Algorithms, Multilateral Bus, SoC, Resource Scheduling

1 Introduction

With the rapid development of system-on-chip (SoC) and multicore processor technologies, the resource contention problem has become a key challenge in the design of multicore systems and network-on-chip (NoC). In these systems, multiple processing units share resources such as buses, memories, and network switches, and efficient scheduling and allocation of resources is critical for system performance improvement. With the increase in computation demand and data transmission, traditional arbitration mechanisms gradually reveal their limitations in dealing with dynamic loads and fluctuating resource demands. Therefore, how to achieve efficient resource allocation

J. Iqbal (ed.), *Proceedings of the 2025 2nd International Conference on Mechanics, Electronics Engineering and Automation (ICMEEA 2025)*, Advances in Engineering Research 272,
https://doi.org/10.2991/978-94-6463-821-9_50

in scenarios with uneven resource demand and large load fluctuations has become one of the hot topics in current research.

Traditional arbitration mechanisms, such as fixed-priority arbitrators and rotation arbitrators (RRAs), were widely used in early multicore processor designs. Fixed-priority arbitrators schedule resources by statically assigning a priority to each request, which is simple and easy to implement, but can easily lead to 'starvation' of low-priority tasks when the load fluctuates or the requests are uneven [1]. The rotation arbiter can effectively avoid starvation by periodically taking turns to allocate resources for each request, but it still cannot make flexible adjustments according to the actual resource demand of tasks, especially in complex systems with large fluctuations in resource demand, its efficiency and fairness will be greatly reduced [2].

With the increasing demands on system performance, researchers are turning to dynamic arbiter design based on evolutionary algorithms. Evolutionary algorithms (e.g., genetic algorithms, ant colony algorithms, etc.) are able to automatically adjust and optimise task priorities and resource allocation strategies by mimicking the process of natural selection, thus demonstrating higher adaptability and efficiency than traditional methods in the face of dynamic loads and variable resource demands. The Evolutionary Arbiter (EVOA) is able to guarantee the bandwidth utilisation and throughput of the system in dynamic environments and avoid long time starvation of low-priority tasks by continuously adjusting the resource allocation strategy [3][4].

However, despite the significant theoretical advantages of arbiters based on evolutionary algorithms, their practical applications still face several challenges. Firstly, the high computational complexity of evolutionary algorithms, especially when dealing with large-scale systems, and the iterative process of evolutionary algorithms may consume a large amount of computational resources, which makes them difficult to implement in hardware. Secondly, how to balance the adaptability and computational efficiency of evolutionary algorithms and reduce the power consumption and area overhead in hardware implementation remains a difficulty in current research [5]. Finally, although the evolutionary arbiter has made some breakthroughs in scheduling performance, how to further improve the efficiency of the algorithm and apply it to a wider range of system scenarios is still the focus of future research.

The purpose of this paper is to review the current state of research on evolutionary arbiters based on evolutionary algorithms in recent years, focusing on the design and application of adaptive weight evolution mechanisms. Firstly, the article will review the basic principles of evolutionary arbiter and its development history, and analyse its application in multi-core processors and network-on-chip systems. Secondly, the paper will categorise and summarise the existing research methods, discuss their advantages and disadvantages as well as the challenges they face. Finally, this paper will look into the future development trend, and propose the research direction of evolutionary algorithm-based arbiters in improving computational efficiency, optimising hardware implementation, and coping with the demands of more complex systems.

The purpose of this paper is to review the current state of research on evolutionary arbiters based on evolutionary algorithms in recent years, focusing on the design and application of adaptive weight evolution mechanisms. Firstly, the article will review the basic principles of evolutionary arbiter and its development history, and analyse its application in multi-core processors and network-on-chip systems. Secondly, the paper will categorise and summarise the existing research methods, discuss their advantages

and disadvantages as well as the challenges they face. Finally, this paper will look into the future development trend, and propose the research direction of evolutionary algorithm-based arbiters in improving computational efficiency, optimising hardware implementation, and coping with the demands of more complex systems.

2 Domestic and International Research Progress

With the rapid development of system-on-chip (SoC) and multi-core processor technologies, how to efficiently manage and schedule these shared resources has become an important topic. Traditional arbitration mechanisms, although effective in some simple scenarios, cannot effectively cope with fluctuations in resource demand and dynamic changes in task priorities when facing dynamic loads and complex applications. Therefore, evolutionary arbitrators based on evolutionary algorithms have gradually become a new research direction.

2.1 International research progress

In the international arena, research on evolutionary arbiters based on evolutionary algorithms began in the late 1990s. The initial evolutionary arbiter design mainly relied on classical evolutionary algorithms such as genetic algorithms (GA), which use the fitness function to evaluate the advantages and disadvantages of resource scheduling schemes, and continuously optimise the arbitration strategy through operations such as selection, crossover and mutation. Related research has focused on how to improve the adaptability of the algorithms, optimise the efficiency of resource scheduling, and reduce the computational complexity.

The combination of evolutionary algorithms and arbiter design Early research on evolutionary arbiters focused on designing more adaptive algorithms, such as genetic algorithm-based evolutionary arbiters. Researchers have proposed a variety of evolutionary algorithm variants for efficient resource allocation and load balancing in multi-core processors and on-chip networks. In 1999, Smith et al. proposed a resource scheduling model based on genetic algorithms to optimise task scheduling by simulating the natural selection process [3]. This study showed that evolutionary algorithms can flexibly handle load fluctuations and task priority changes to optimise system performance.

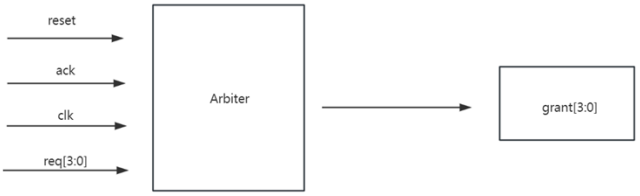

Fig. 1. Block diagram of the bus arbiter [3]

Fig 1 is a block diagram of a bus arbiter, showing the architecture of a bus arbiter used for four master devices. It can be used to illustrate the basic working principle of the bus arbiter.

Along with the introduction of adaptive weight adjustment mechanism, in recent years, researchers have gradually introduced adaptive weight adjustment mechanism into evolutionary arbiter design to better cope with dynamic resource demands. The adaptive mechanism can automatically adjust the priority of each task according to the real-time load situation, thus achieving flexible scheduling of resources. For example, Huang proposed an arbiter design based on an adaptive evolutionary algorithm, which improves the bandwidth utilisation and throughput of the system by dynamically adjusting the priority weights of tasks. Hardware implementation and real-time challenges, the computational complexity of the evolutionary arbiter is high, how to implement it in hardware and improve its real-time has become the focus of research. Hassani investigated the hardware implementation problem of evolutionary algorithms in network switching, and proposed an evolutionary arbiter architecture for on-chip networks, and optimised the algorithm's computational efficiency and reduced the power consumption through hardware acceleration techniques [6].

2.2 Progress of domestic research

Domestic research on evolutionary arbiters based on evolutionary algorithms started late, but in recent years, with the rapid development of multi-core processors and on-chip network technology, related research has gradually made some breakthroughs. Domestic research mainly focuses on algorithm optimisation, hardware implementation and application in multilayer bus systems.

Resource scheduling optimisation based on evolutionary algorithms, domestic scholars have made several innovations in the design of arbiters based on evolutionary algorithms. For example, in 2016, Zhang Qiang et al. proposed an improved genetic algorithm for optimising resource scheduling in SoCs, which improved resource utilisation by introducing an adaptive weighting mechanism [7]. This study showed that the genetic algorithm can effectively adjust task priorities under dynamic load conditions, thus improving the overall performance of the system.

Research and Application of Adaptive Weighting Mechanism Domestic researchers have also made significant contributions to the design of adaptive weighting mechanism. Zhu Yaqi proposed an evolutionary arbiter design based on adaptive weighting, which dynamically adjusts the priority of tasks using an improved ant colony algorithm, which can effectively avoid the starvation problem of low-priority tasks and improve the bandwidth utilisation of the system-on-chip [4]. The design is applied in a multilayer bus system, and the experimental results show that the evolutionary arbiter can effectively cope with the situation of large fluctuations in resource. demand

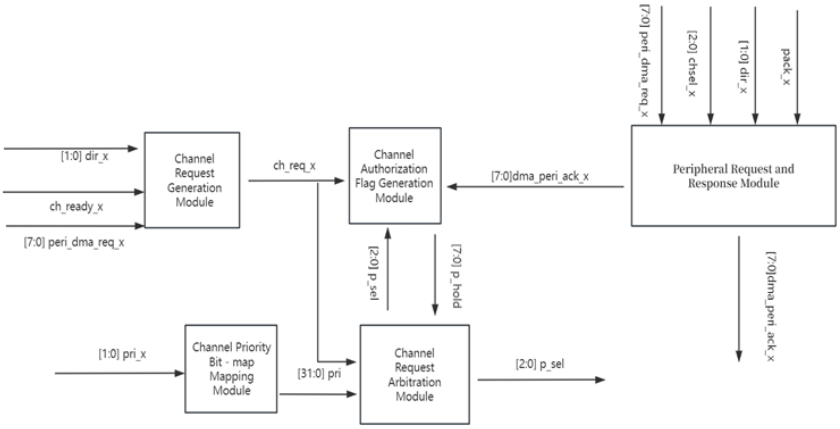

Fig. 2. Block diagram of DMA data stream arbiter hardware architecture [4]

The design of the DMA request response module and the request arbitration mechanism has received extensive attention in domestic research. Some domestic researches draw on the priority scheduling and DMA request response mechanism in the AMBA bus protocol, such as the peripheral DMA request response module and the channel request generation module demonstrated by Yaqi Zhu in Fig 2. These modules achieve effective scheduling of multiple DMA requests through priority control and bit-map mapping techniques, ensuring fair allocation and efficient use of system resources. Specifically, the Peripheral DMA Request Response Module is responsible for receiving and responding to DMA requests from peripherals, while the Channel Request Generation Module generates channel request signals by processing incoming request signals and schedules them flexibly according to the system load.

In addition, the design of request arbitration module and channel authorisation generation module has been widely used in the multi-core system and SoC design in China. By introducing the priority bit-map mapping and channel authorisation mechanism, these designs effectively solve the resource contention problem in the multi-task parallel environment, avoiding the phenomenon that some requests are 'starved' due to low priority. The request arbitration module dynamically adjusts the task priority to ensure the fairness of the system, while the channel authorisation generation module determines which request can obtain bus access according to the scheduling policy, which greatly improves the efficiency of system resources.

In practical applications, some domestic researches have also combined the design of these modules with multi-core processors and system-on-chip (SoC) to optimise the efficiency of data transmission. The flexibility and configurability of these modules have been fully verified in different application scenarios, especially in high-performance systems that need to handle large-scale data transmission, which can effectively improve throughput and reduce latency.

Hardware implementation and low-power optimisation. Domestic research has also made active exploration in the hardware implementation of evolutionary arbiters.

Zhisheng Xu proposed a dynamic weight arbiter design based on on-chip network, which optimises the computational efficiency of the algorithm using low-power hardware implementation and achieves a better balance between real-time and resource consumption [5][8]. This research provides theoretical and practical support for the practical application of evolutionary arbiters, especially in complex systems that require real-time resource scheduling.

3 Adaptive weighting with evolutionary mechanisms

In recent years, with the increase of system complexity and the prominence of the resource contention problem, the traditional static arbitration mechanism gradually fails to meet the demand for resource scheduling in high-performance systems. The combination of adaptive weighting and evolutionary mechanism provides a new idea to solve this problem. By combining the evolutionary algorithm and the dynamic scheduling mechanism, the system is able to adjust the priority of tasks and the resource allocation strategy in real time, thus achieving more efficient and fair resource scheduling.

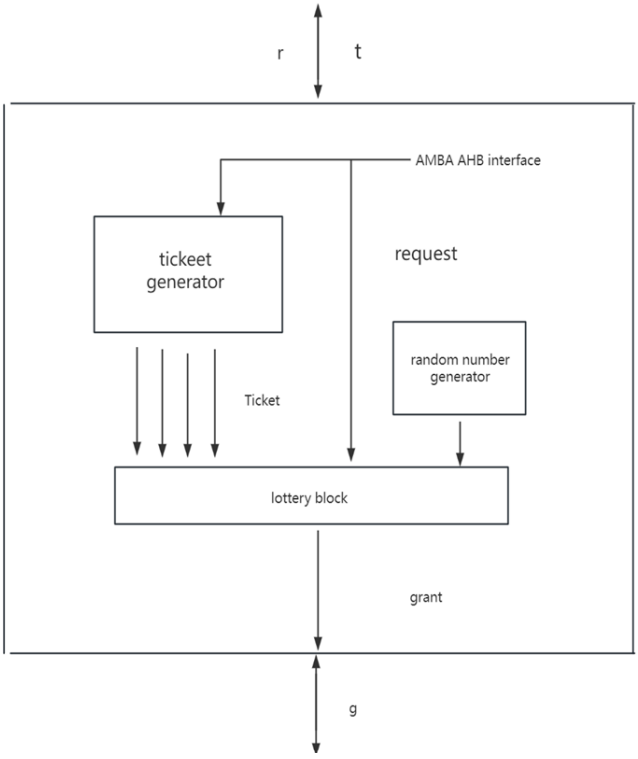

Fig. 3. Schematic diagram of the dynamic adaptive arbiter structure [9]

Fig 3 shows the schematic structure of a dynamic arbiter, which is widely used in

multi-request environments where fair and unbiased requests are required, especially in systems with limited resources and multiple request sources competing for resources at the same time. Due to its randomized nature, the dynamic arbiter is able to avoid the situation where some requests are not resourced for a long time due to a fixed priority policy.

3.1 Fundamentals of the adaptive weighting mechanism

Adaptive weighting mechanism means that the system dynamically adjusts the priority of tasks according to the real-time load situation and the resource demand of request sources. This mechanism enables the arbiter to flexibly allocate resources according to different request sources and load changes of the system. For example, when the load is light, low-priority tasks may be prioritized for resources, whereas when the system load is heavier, the resource demands of high-priority tasks will be satisfied first.

Unlike traditional static priority arbiters, the adaptive weighting mechanism enables the system to respond quickly to load fluctuations and resource demand changes by adjusting the weights in real time [1]. The advantage of this mechanism is that it can balance the resource allocation of each task according to the dynamic demand of the system, avoiding the phenomenon of task starvation and improving the resource utilization.

3.2 Application of evolutionary algorithms in resource scheduling

Evolutionary algorithms (e.g. genetic algorithms, ant colony algorithms, etc.) simulate the process of natural selection, and gradually optimize the resource allocation strategy of the system through continuous selection, crossover and mutation operations. In environments with large dynamic load changes, evolutionary algorithms can dynamically adjust the resource scheduling strategy according to the system's historical data and real-time feedback, thus realizing optimal resource management.

The evolutionary algorithm continuously improves the resource allocation scheme through multiple iterations, and is able to adaptively adjust the priority of tasks and select the most appropriate resource scheduling scheme. This scheduling method based on evolutionary algorithms has strong adaptability in complex systems and is especially suitable for multi-task and multi-request environments [7].

In the design of the AMBA bus protocol, the adaptive weighting mechanism combined with the evolutionary algorithm can optimize the resource allocation in the multi-master device environment, and improve the bandwidth utilization and throughput by dynamically adjusting the request priority. This scheduling method not only adapts to load fluctuations, but also avoids the starvation problem of low-priority tasks under high system load.

3.3 Combination of adaptive weights and evolutionary mechanisms

The adaptive evolutionary mechanism formed by combining the adaptive weighting mechanism with the evolutionary algorithm has obvious advantages. In this mechanism, the evolutionary algorithm continuously optimizes the resource allocation

strategy by evaluating the adaptability of multiple resource scheduling schemes, while the adaptive weighting mechanism ensures that tasks are dynamically adjusted in terms of priority according to the real-time load situation.

This combination not only improves the system performance, but also ensures the fairness of the system. Through the optimization of evolutionary algorithms, the system is able to find the best resource scheduling scheme; while through the adaptive weighting mechanism, the system is able to flexibly adjust the resource allocation strategy to ensure the timely execution of high-priority tasks, while avoiding low-priority tasks from being ignored for a long time.

In several literatures, the design combining evolutionary algorithm and adaptive weighting mechanism can significantly improve the flexibility and efficiency of resource scheduling, especially in the application of multi-tier bus system and data center [8].

3.4 Application of Adaptive Evolutionary Mechanisms in Multicore Systems

In a multicore system, multiple processor cores share resources, and how to efficiently schedule these resources is an important issue in design. A mechanism combining adaptive weights and evolutionary algorithms can efficiently manage resource requests from multiple cores in a multicore system, avoiding resource contention and improving system performance. By introducing evolutionary algorithms, the system can schedule resources based on load fluctuations and task priorities in real time, ensuring that each core can access shared resources fairly and efficiently [7].

In some high-performance computing and real-time systems, adaptive evolutionary mechanisms are particularly suitable for multi-task scheduling. By introducing an evolutionary algorithm into the arbiter, the system can automatically optimize the scheduling strategy according to the actual situation, so as to avoid the over-occupation of resources and the delay of high-priority tasks.

4 Design Challenges of SoC and Multilayer Buses

With the wide application of SoC (System-on-Chip) technology, the integration of multiple cores and peripherals makes system design more complex. In multilayer bus architectures, multiple master devices share bus resources, and how to effectively manage these shared resources has become a key issue to improve SoC performance [10]. In this environment, traditional arbitration strategies (e.g., rotation arbitration, priority arbitration, etc.) face several challenges, especially in multitasking and task-dependent application scenarios, where how to achieve fine-grained bandwidth control and fair resource allocation becomes an urgent problem.

4.1 Task dependencies and bandwidth control challenges

In SoC systems, multiple applications often have different task dependencies, which means that some tasks have to be executed after other tasks have been completed. This complicates the resource management of the system, as bandwidth allocation must not

only take into account the dependencies between tasks, but also avoid resource contention. In existing arbitration mechanisms, such as Round-Robin and Weighted Round-Robin, it is usually difficult to deal with this dynamic allocation of bandwidth efficiently because these methods fail to fully consider inter-task dependencies and changes in bandwidth requirements [10].

For example, Round-Robin methods, despite ensuring fairness, lack fine-grained control over bandwidth, especially in application scenarios with complex task dependencies, which can lead to long task waiting times for some applications and degrade the overall system performance. Similarly, Weighted Round-Robin, although it takes into account the priority of tasks, still struggles to cope with the challenges of bandwidth management under dynamic loads and changes in task dependencies due to its reliance on fixed-weight allocation.

4.2 Application of evolutionary mechanisms and adaptive weights

In order to meet the challenges of bandwidth control and task scheduling, the combination of adaptive weighting and evolutionary mechanism is proposed as a possible solution. Through the adaptive weighting mechanism, the bandwidth allocation for each request can be dynamically adjusted according to the real-time load of the system and the priority of the task, thus improving the resource utilisation and avoiding task 'starvation'. For example, Supervised Debt Opportunistic (SuDO) is a novel arbitration policy that combines debt monitoring and opportunistic access to achieve fine-grained control of bandwidth, thus avoiding the shortcomings of traditional arbitration policies, in fig 4 [10].

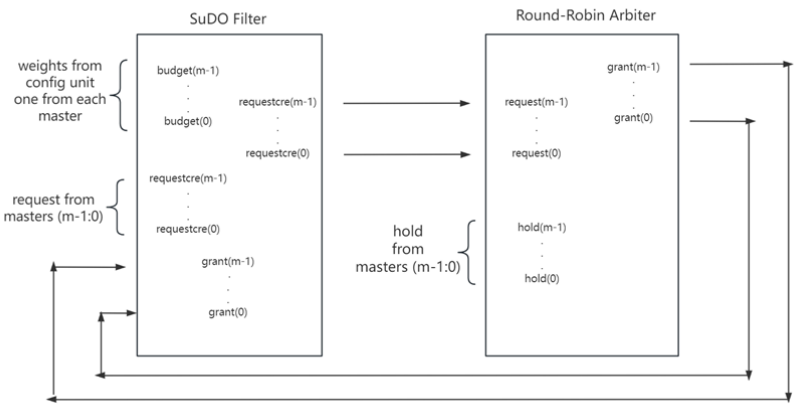

Fig. 4. SUDO Arbiter architecture [10]

4.3 The need to achieve granular bandwidth control

With the increase of task dependency and load fluctuation, the design of SoC systems requires more flexible and efficient arbitration strategies to ensure the fair and efficient use of resources. By combining evolutionary algorithms and adaptive scheduling mechanisms, the system is able to adjust the allocation of resources in real time according to different task requirements and priorities, avoid over-allocation of bandwidth to certain master devices, and ensure that each request can obtain the required resources in a timely manner.

In order to effectively support multi-tasking applications and avoid bandwidth waste, the new arbitration strategy needs to ensure that different types of tasks can be allocated bandwidth on demand, especially in multi-layer bus systems, where the tasks and bandwidth requirements of each master device may be different, so the system's bandwidth scheduling strategy must have a high degree of flexibility and adaptability.

5 Conclusion

With the rapid development of System-on-Chip (SoC) technology, especially in the environment of multi-core processors and peripheral integration, resource management and scheduling become critical issues. Traditional arbitration methods (e.g., priority arbitration, rotation arbitration) are difficult to cope with resource scheduling in multi-task and dynamic load situations. For this reason, bandwidth-controlled arbitrators and dynamic resource scheduling mechanisms have gradually become effective solutions to this problem. In this paper, we reviewed the bandwidth control arbitration mechanism based on adaptive weights and evolutionary algorithms, and explored how to optimise the resource utilisation and performance of multi-core SoC systems by dynamically adjusting task priorities and bandwidth allocation. Future research can focus on the following directions: first, with the increase in the number of cores, the existing bandwidth control mechanism needs to be extended to support efficient resource scheduling for more devices; second, scheduling algorithms based on deep and reinforcement learning can automatically learn from the load changes and optimise the resource allocation to enhance the scheduling flexibility and efficiency; third, how to ensure the performance of the system while ensuring the system performance through hardware optimisation and low-power scheduling strategies to reduce energy consumption is an important direction in future design; finally, with the coexistence of multiple bus protocols, how to design resource scheduling mechanisms that support multiple protocols becomes a key to future research. The bandwidth control arbiter based on evolutionary algorithm and adaptive weighting mechanism demonstrates powerful resource scheduling capability in multi-core SoCs. As the system complexity increases, future research will further improve the efficiency, flexibility and energy efficiency of the scheduling mechanism to meet the demands of larger scale and complex tasks.

References

1. Helal, K. A., Attia, S., Ismail, T., Mostafa, H.: 'Priority-select arbiter: An efficient round-robin arbiter', 2015 IEEE 13th International New Circuits and Systems Conference (NEWCAS), 2015, pp. 1–4
2. Vadayath, J., Eckert, M., Zeng, K., et al.: 'Arbiter: Bridging the static and dynamic divide in vulnerability discovery on binary programs', 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 413–430
3. Mooney, V. J., Riley, G. F., Shin, E. S.: 'Round-robin Arbiter Design and Generation', 15th International Symposium on System Synthesis, 2002, pp. 243–248
4. Zhu, A., Hou, X.-J.: 'Design of DMA data stream arbiter based on bit-map algorithm', Microcontroller and Embedded System Applications, 2023, 9, pp. 74–77+82
5. Xu, C. S.: 'Design of Dynamic Weight Arbiter Based on Chip Network Circuit', Journal of Changchun Engineering College (Natural Science Edition), 2019, 4, pp. 55–58+86
6. Aguénounon, E., Razavinejad, S., Schell, J. B., et al.: 'Design and characterization of an asynchronous fixed priority tree arbiter for SPAD array readout', Sensors, 2021, 21, (12), pp. 3949
7. Ji, Y. K.: 'A dynamic allocation arbiter design based on multilayer bus', Modern Computers, 2020, 5, pp. 105–108
8. Singh, A. K., Shrivastava, A., Tomar, G. S.: 'Design and implementation of high performance AHB reconfigurable arbiter for on-chip bus architecture', 2011 International Conference on Communication Systems and Network Technologies, 2011, pp. 455–459
9. Xu, Y., Li, L., Du, G. M., et al.: 'A dynamic adaptive arbiter based on multiprocessor system-on-chip', Computer Research and Development, 2008, 6, pp. 1085–1092
10. Ibarra-Delgado, S., Sandoval-Arechiga, R., Gómez-Rodríguez, J. R., et al.: 'A Bandwidth Control Arbitration for SoC Interconnections Performing Applications with Task Dependencies', Micromachines, 2020, 11, (12), pp. 1063