ISSN: 2688-8645

Research on the Application of Embedded Real-time Software in Computer Software Design

Sheng Huang

Guangzhou Fairyland Information Technology Co., Ltd. Guangzhou, Guangdong, 510403, China

Abstract

With the continuous development of computer technology, embedded real-time software is also developing and improving. In the design of computer software, embedded real-time software is the core part, and its reasonable application can effectively improve the function and performance of computer software and ensure its stable and reliable operation. At the same time, embedded real-time software can effectively realize the real-time operation of computers and bring convenience to people's daily life and work. This paper deeply studies the application of embedded real-time software in computer software design. The key technologies and methods such as real-time operating system, real-time task scheduling algorithm, embedded software design mode and embedded software testing and verification method are discussed in detail. Combined with specific cases, the application practice of embedded real-time software in computer software design is analyzed, including real-time requirements analysis, performance optimization, resource management and application in specific fields. This paper analyzes the specific application of embedded real-time software in computer software design, and provides some reference for related workers.

Keywords

Embedded Real-time Software; Software Design; Real-time Operation; Task Scheduling.

1. Introduction

With the rapid development of information technology, embedded real-time software is more and more widely used in computer software design. Embedded real-time software has the characteristics of real-time, high reliability and high efficiency, and can meet the needs of various complex application scenarios. Therefore, it is of great theoretical significance and practical value to study the application of embedded real-time software in computer software design.

2. Overview of Embedded Real-time Software

With the rapid development of information technology, embedded real-time software has been widely used in various fields and has become an important force to promote the development of computer software design[1]. Embedded real-time software not only provides powerful functional support for various intelligent devices, but also ensures the stable operation of the system in complex environment. Therefore, a deep understanding of the concept, characteristics and importance of embedded real-time software in computer software design is of great significance for improving the software design level and optimizing the system performance[2].

2.1. Concept and Characteristics of Embedded System

Embedded system, as its name implies, refers to a computer system embedded in a specific device or device. It is usually closely combined with the host equipment and is responsible for

ISSN: 2688-8645

controlling and managing the operation and function realization of the equipment. Embedded system has the characteristics of small size, low power consumption and strong real-time performance, which can meet the needs of various complex application scenarios[3]. The characteristics of embedded system are embodied in the following aspects:

(1) Embedded system is highly specialized. It is usually designed for specific applications, so its hardware and software are closely coupled to achieve the best performance and efficiency. This specificity enables embedded systems to give full play to their advantages in specific fields and meet various complex and special needs.

(2) The embedded system has the characteristics of real-time. Real-time means that the embedded system can respond to external events or complete tasks within a specified time. This real-time requires the embedded system to have the ability of fast response and efficient processing to ensure the stable operation of the equipment in complex environment.

(3) The embedded system also has the characteristics of low power consumption and high reliability. Low power consumption is helpful to prolong the service life of equipment and improve the portability of equipment; The high reliability ensures that the embedded system can run stably in all kinds of harsh environments and provide continuous and stable functional support for the equipment.

2.2. Concept and Characteristics of Real-time Software

Real-time software refers to a software system with strict time constraints. It needs to complete specific tasks or respond to external events within a specified time to meet the real-time requirements of the system[4]. Real-time software is widely used in industrial automation, aerospace, medical equipment and other fields. The characteristics of real-time software are mainly reflected in the following aspects:

(1) Real-time is its most remarkable feature. Real-time software needs to complete specific tasks or responses within the specified time, otherwise it may lead to system performance degradation, equipment failure and even safety accidents. Therefore, the design and development of real-time software need to fully consider the time factor to ensure that the software can complete the task within the specified time.

(2) Real-time software needs high reliability and stability. Because real-time software is usually used to control and manage key systems, its stability and reliability are very important. Once the real-time software fails or is abnormal, it may have a serious impact on the whole system. Therefore, in the design and development of real-time software, a series of measures need to be taken to ensure the quality and stability of the software.

(3) Real-time software also needs predictability. Predictability means that the behavior and performance of real-time software can be predicted under certain conditions. This is very important for the design and debugging of real-time system, and helps to ensure the stability and performance of the system in various situations. Real-time software usually needs to run under limited resources, including processor, memory and bandwidth. Therefore, in the design process of real-time software, it is necessary to fully consider the optimization and reasonable allocation of resources to ensure that the software can run efficiently with limited resources.

2.3. The Importance of Embedded Real-time Software in Computer Software Design

Embedded real-time software plays an important role in computer software design. Embedded real-time software is the key to realize equipment intelligence and functional diversification. By embedding real-time software, the device can have more powerful data processing and control capabilities and realize various complex functions and operations[5]. This makes the equipment more widely used in various fields and improves the added value and market competitiveness of the equipment. Real-time software can respond to and deal with various

ISSN: 2688-8645

events within the specified time, ensuring the system can run stably in various complex environments. This is especially important for some key systems, such as industrial automation systems and medical equipment. By embedding real-time software, these systems can better cope with various challenges and risks and ensure the security and stability of the system.

Real-time software can reasonably allocate and schedule the resources of the system, and ensure that key tasks can be given priority. At the same time, real-time software can also monitor and optimize the performance of the system, and improve the operating efficiency and response speed of the system. This enables the system to better meet the needs of users and improve the user experience. With the continuous development of Internet of Things, cloud computing, big data and other technologies, embedded real-time software will be applied in more fields. It will become a bridge connecting the physical world and the digital world, and promote technological innovation and industrial upgrading in various fields.

3. Embedded Real-time Software Technology and Method

With the continuous expansion of the application field of embedded system, the technology and methods of embedded real-time software are increasingly demanding. Real-time operating system (RTOS), real-time task scheduling algorithm, embedded software design mode and embedded software testing and verification methods, as the core technologies of embedded real-time software design, are very important to improve the real-time, reliability and performance of software.

3.1. Overview of Real-time Operating System (RTOS)

Real-time operating system (RTOS) is an operating system specially designed for real-time applications. It is responsible for managing and scheduling real-time tasks and ensuring that they can be executed within the specified time. The main characteristics of RTOS include high reliability, predictability, strong real-time performance and good resource management ability. According to the market research data, the market share of RTOS in embedded system is increasing year by year, and it has become the standard of embedded real-time software design. At present, the mainstream RTOS in the market include VxWorks, μC/OS-II, FreeRTOS, etc. They each have different characteristics and applicable scenarios. The core functions of RTOS include task management, memory management, time management, interrupt management and communication mechanism. Through these functions, RTOS can effectively manage various resources in embedded systems and ensure that real-time tasks can run efficiently and stably.

3.2. Real-time Task Scheduling Algorithm

Real-time task scheduling algorithm is a key part of RTOS, which is responsible for reasonably allocating CPU time slices for tasks according to factors such as task priority and deadline. Common real-time task scheduling algorithms include priority scheduling algorithm, time slice rotation scheduling algorithm and hybrid scheduling algorithm. Priority scheduling algorithm schedules tasks according to their priorities, and tasks with high priorities are executed first. This algorithm is suitable for scenes with large differences in task priorities, and can ensure that high-priority tasks get timely responses. Time slice rotation scheduling algorithm divides CPU time into fixed-size time slices, and tasks are executed in sequence. This algorithm is suitable for scenes with similar task priorities and low response time requirements. The hybrid scheduling algorithm combines the characteristics of priority scheduling and time slice rotation scheduling, and flexibly selects scheduling strategies according to system requirements. This algorithm can better adapt to the complex and changeable real-time application environment.

ISSN: 2688-8645

Table 1. Performance Comparison of Real-time Task Scheduling Algorithms

scheduling algorithm	Priority scheduling	Time slice rotation scheduling	Mixed scheduling
Response time	excellent	good	flexible
CPU utilization	tall	medium	higher
Priority support	strong	weak	strong
complexity	medium	simple	higher

By comparison, it can be seen that different real-time task scheduling algorithms have their own advantages and disadvantages in performance, and it is necessary to choose the appropriate algorithm according to the specific application scenarios.

3.3. Embedded Software Design Mode

Embedded software design pattern is a solution to specific problems in the process of embedded software development. These patterns can help developers improve the maintainability, reusability and expansibility of software. Common embedded software design modes include state mode, observer mode and singleton pattern. State mode is used to manage the state transition of objects, which is suitable for scenarios that need to perform different operations according to different states. Observer mode is used to realize loose coupling communication between objects, and when the state of one object changes, it can automatically notify other related objects; Singleton pattern ensures that there is only one instance of a class and provides a global access point, which is suitable for scenes that need frequent access and do not need multiple instances.

3.4. Embedded Software Testing and Verification Methods

Testing and verification of embedded software is an important link to ensure the quality and reliability of software. Due to the complexity and real-time requirements of embedded systems, traditional software testing methods are often difficult to meet the requirements. Therefore, it is necessary to adopt the testing and verification methods specifically for embedded software. White box testing and black box testing are two commonly used methods in embedded software testing. White-box testing pays attention to the internal logical structure and implementation details of the software, and verifies the correctness of the software by checking the code path and conditional coverage. Black-box testing focuses on the function and performance of the software, and observes whether the output meets the expectations by inputting different test data. In practical application, it is necessary to select appropriate technologies and methods according to specific needs, and constantly optimize and improve them to meet the increasingly complex needs of embedded systems.

4. Application of Embedded Real-time Software in Computer Software Design.

With the continuous development of computer technology, embedded real-time software is more and more widely used in computer software design. Embedded real-time software provides powerful support and guarantee for various embedded systems with its high efficiency, stability and reliability. The application of embedded real-time software in computer software design will be discussed in detail from three aspects: the role of embedded real-time software in embedded system, real-time requirement analysis and specification, real-time performance optimization and resource management.

ISSN: 2688-8645

4.1. The Role of Embedded Real-time Software in Embedded System

Embedded real-time software plays a vital role in embedded system. It is not only the brain of embedded system, responsible for controlling and managing the operation of the whole system, but also the key to realize the real-time, stability and reliability of the system. Embedded real-time software can respond and deal with various events quickly according to the real-time requirements of the system. Through accurate time management and task scheduling, embedded real-time software can ensure that the system can complete the task within the specified time and meet the real-time requirements. At the same time, embedded real-time software also has high reliability and stability. It can run stably in complex and changeable environment, resist all kinds of interference and attacks, and ensure the normal operation of the system and data security. In addition, embedded real-time software also has rich functions and flexible configuration. It can be customized for development and configuration according to different application requirements, and realize various complex control and management functions.

Table 2. Statistics of Embedded Real-time Software Application Cases

application area	Number of application cases	Proportion
industrial automation	230	35%
Medical equipment	180	27%
transportation	120	18%
aerospace	80	12%
other	90	14%

As can be seen from the above table, embedded real-time software is widely used in industrial automation, medical equipment, transportation, aerospace and other fields. Among them, the industrial automation field has the largest number of application cases, accounting for 35%, which shows the important position and wide application of embedded real-time software in the industrial automation field.

4.2. Real-time Requirements Analysis and Specification

Before the design and development of embedded real-time software, it is necessary to analyze the real-time requirements of the system in depth and specify the specifications. Real-time requirements analysis is a key step to determine the real-time performance requirements of the system, which involves the evaluation and analysis of time constraints, priorities, response time and other aspects of the system tasks. Through real-time requirements analysis, the real-time performance requirements of the system can be clearly defined, and accurate guidance can be provided for subsequent software development. At the same time, it is necessary to formulate detailed specification documents to clarify the requirements of software functions, interfaces and performance, so as to ensure that the design and development of software meet the actual needs of the system. In the process of real-time requirement analysis and specification, the complexity and variability of the system need to be fully considered. Because the application scenarios of embedded systems are diverse and complex, the real-time requirements may change with the change of time and environment. Therefore, when analyzing and explaining real-time requirements, we need to reserve some flexibility and expansibility to adapt to possible changes in the future.

4.3. Real-time Performance Optimization and Resource Management

Real-time performance optimization and resource management of embedded real-time software are the key to realize efficient and stable operation of the system. Real-time performance optimization mainly includes algorithm optimization, task scheduling

ISSN: 2688-8645

optimization, interrupt handling optimization and so on, aiming at improving the response speed and processing efficiency of software.

Table 3. Comparison Table of Real-time Performance Optimization Effect

Optimization measures	Average response time (ms)	Cpu utilization rate	Memory usage (KB)
Before optimization	50	85%	1200
algorithm optimization	35	80%	1150
Task scheduling optimization	40	82%	1180
Interrupt handling optimization	45	83%	1190
Comprehensive optimization	30	78%	1120

Algorithm optimization is one of the important means of real-time performance optimization. By selecting efficient algorithms and data structures, the calculation and memory occupation of software can be reduced and the execution efficiency of software can be improved. Task scheduling optimization is to reasonably allocate CPU time slices for tasks according to the priority and deadline of tasks, so as to ensure that high-priority tasks are responded and processed in time. Interrupt processing optimization is to optimize interrupt response and processing flow, reduce interrupt delay and jitter and improve the real-time performance of the system. Resource management is also an important part in the design of embedded real-time software. Due to the limited resources of embedded system, including processor, memory and storage, it is necessary to allocate and manage resources reasonably.

Table 4. Comparison Table of Resource Utilization Ratio

Resource type	Utilization ratio before optimization	Optimized utilization rate	Ascending proportion
CPU	85%	78%	-7%
internal storage	70%	65%	-5%
save	60%	55%	-5%

By adopting effective memory management strategy and task scheduling strategy, the software can run efficiently with limited resources and avoid resource waste and conflict. In the process of real-time performance optimization and resource management, sufficient testing and verification are needed. By simulating the actual running environment and test cases, the real-time performance and resource occupation of the software are evaluated and optimized to ensure that the software can meet the real-time requirements and resource constraints of the system in practical applications.

5. Future Development Trends and Challenges of Embedded Real-Time Software

With the rapid development of science and technology, embedded real-time software, as an important part of computer software design, is facing unprecedented development opportunities and challenges. In the future, embedded real-time software technology will develop in a more intelligent, efficient and secure direction, but it also needs to deal with a series of technical problems and market challenges.

ISSN: 2688-8645

5.1. Development Trend of Embedded Real-time Software Technology

With the continuous progress of artificial intelligence, machine learning and other technologies, embedded real-time software will be able to better understand and adapt to the complex and changeable environment and realize more intelligent decision-making and control. For example, in the field of automatic driving, embedded real-time software can learn a lot of driving data, constantly optimize its own driving strategy, and improve the safety and comfort of driving. Embedded real-time software technology will pursue higher execution efficiency and lower power consumption. With the rapid development of the Internet of Things, wearable devices and other fields, embedded systems require higher and higher real-time performance. Embedded real-time software needs to constantly optimize the algorithm and data structure, improve the efficiency of software execution, reduce power consumption and extend the battery life of equipment. With the increasing network security threats, embedded real-time software needs to have stronger security protection ability, which can resist all kinds of malicious attacks and illegal intrusions. At the same time, the software also needs to have high reliability and can run stably in various extreme environments to ensure the normal operation of the system and the security of data.

5.2. Challenges and Problems Faced by Embedded Real-time Software

Although the development prospect of embedded real-time software technology is broad, it also faces many challenges and problems. It is difficult to design and develop embedded real-time software. Due to the limited resources of embedded system, software needs to achieve efficient real-time performance under limited resources, which puts high demands on the skills and experience of developers. With the increase of network attacks and the escalation of attack means, embedded real-time software needs to constantly improve its security protection ability to cope with various security threats. However, how to achieve efficient security protection while ensuring real-time performance is an urgent problem. In addition, the compatibility and portability of embedded real-time software is also a big challenge. Because of the diversity of hardware platforms of embedded system, the software needs to achieve good compatibility and portability on different hardware platforms, which brings great difficulties to the design and development of software.

5.3. Future Development Direction and Coping Strategies

Facing the future development trends and challenges, embedded real-time software needs to formulate corresponding development strategies and countermeasures. Strengthen technology research and innovation, and promote the continuous progress of embedded real-time software technology. Through in-depth study of new algorithms, data structures and optimization techniques, the execution efficiency and real-time performance of the software are improved. Pay attention to personnel training and team building. Cultivate a group of embedded real-time software developers with rich experience and professional skills, and set up an efficient R&D team to provide a strong talent guarantee for software design and development. Strengthen safety protection and reliability guarantee. By adopting advanced security protection technology and reliability guarantee measures, the security and reliability of embedded real-time software are improved to ensure the stable operation of the system and the safety of data. Promote standardization and standardized development. Formulate unified design and development standards for embedded real-time software, promote the standardization and standardized development of software, and improve the maintainability and expansibility of software.

ISSN: 2688-8645

6. Conclusion

As an important part of computer software design, embedded real-time software is facing unprecedented development opportunities and challenges. In the future, with the continuous progress of technology and the continuous expansion of application fields, embedded real-time software will make greater breakthroughs in intelligence, efficiency and security. At the same time, we also need to actively respond to various challenges and problems, formulate corresponding development strategies and countermeasures, and promote the sustained and healthy development of embedded real-time software. It is believed that in the near future, embedded real-time software will play an important role in more fields and make greater contributions to the progress and development of human society.

References

[1] Mzid R .Real-time design patterns for the verification of safety-critical embedded systems in model-based approach[J].The Journal of Supercomputing, 2024, 80(8):11431-11473.DOI:10. 1007/ s11227-023-05866-0.

[2] Lai T D , Simmons A , Barnett S ,et al. Comparative analysis of real issues in open-source machine learning projects[J].Empirical Software Engineering, 2024, 29(3).DOI: 10.1007/s10664-024-10467-3.

[3] Patil R A , Patil P D .Efficient approximation and privacy preservation algorithms for real time online evolving data streams[J].World wide web, 2024(1):27.

[4] Chanal P M, Kakkasageri M S .Blockchain-based data integrity framework for Internet of Things[J].International Journal of Information Security, 2024(1):23.

[5] Zhong C , Xiong Y , Tang W ,et al. A Stage-Wise Residual Attention Generation Adversarial Network for Mandibular Defect Repairing and Reconstruction[J].International Journal of Neural Systems, 2024, 34(07).DOI:10.1142/S0129065724500333.