Proceedings of the Third International Conference on Electronics and Sustainable Communication Systems (ICESC 2022)
IEEE Xplore Part Number: CFP22V66-ART; ISBN: 978-1-6654-7971-4

2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC) | 978-1-6654-7971-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICESC54411.2022.9885670

Implementation of Embedded Real-Time Operating System and Application Software based on Smart Chip

Xiaofeng Shang
Department of information management and information systems, School of management, City Institute, Dalian University of Technology, Dalian 116600, China
Channelshang2@mail.com

***Abstract*—In this paper, the design of embedded real-time operating system based on computer network technology and smart chip is proposed. In the hardware design, design the micro-kernel architecture, keep the basic services in the system kernel, move other functions out of the user space, design the Ethernet controller, the embedded and tailorable TCP/IP protocol iwip in uc selenium s— The transplantation process on II. Finally, it focuses on the characteristics of μCOS-II and the transplantation process, and studies the software development methods and processes of the NIOSII soft core and embedded real-time operating system μCOS-II.**

Keywords—Embedded, Real-Time Operating System, Application Software, Smart Chip

I. INTRODUCTION

Embedded system is defined as: application-centric, computer technology-based, software and hardware can be tailored, adapted to the application system, special-purpose computer systems with strict requirements on function, reliability [1], cost, volume, and power consumption. The embedded system is the product of the combination of advanced [2] computer technology, semiconductor technology, electronic technology and specific applications in various industries [3]. This determines that it must be a technology-intensive, capital-intensive [4], highly dispersed, and constantly innovative knowledge integration system.

The peripheral equipment of the embedded computer contains multiple embedded microprocessors, such as keyboards [5], hard disks, monitors, network cards, sound cards, etc., which are all controlled by the embedded processors [6]. Now, the annual industrial output value brought by embedded systems has exceeded 1 trillion US dollars, and embedded computers have been widely used not only in civilian products but also in military equipment [7]. With the continuous improvement of the performance of micro-processors, the application range of embedded technology has become wider and wider, and various functional systems developed using embedded technology have been widely used in all walks of life. Based on computer technology [8], according to actual application needs, combined with embedded technology to develop corresponding functions, starting from the two parts of hardware and software, complete the development of the entire embedded system [9]. The system developed by embedded technology is mainly based on computer technology. Through the actual needs of the application, the corresponding functions are developed, and the embedded system is developed by combining [10] the two parts of hardware development and software development and design. With EDA technology and the rapid development of microelectronics [11] technology has accelerated the development of miniaturization and miniaturization of electronic systems, which has brought profound changes to classic design methods [12]. In the past, designers could choose logic devices, processors, and memories to develop embedded systems. However [13], the appearance of FPGA chips completely changed the traditional design method, because a single chip can realize many electronic system functions [14].

Moreover, in the initial stage of design, after the product is completed, and during use, users can add required functions to it according to actual needs [15]. In the past, FPGA chips could only implement some logic functions, or as the logic functions of certain processors [16]; now, the advent of intellectual property (IP) has brought a revolution to FPGAs. Therefore, many commercial FPGA chips use IP cores. By embedding one's own operation unit [17] and amplifier, this kind of overall thinking design method can be realized. The shortcoming at this time is that there is no microprocessor, so embedded operating [18] system cannot be used, so SOPC technology was born. In the initial stage of the development of computer technology [19], there is no concept of operating system in computer systems. In order to provide users with an interface with the computer, and at the same time to improve the resource utilization of the computer, a computer monitoring program appeared [20], allowing users to use the computer through the monitoring program. With the development of computer technology [21], the hardware and software resources of the computer system are becoming more and more abundant, and the monitoring program can no longer meet the requirements of computer applications [22].

So in the mid-century, the monitoring program developed further and formed the operating system [23]. Up to now, there are three kinds of operating systems that are widely used, namely multi-channel batch operating systems, time-sharing operating systems and real-time operating systems. Schiffer et al. researched [24] and produced the indoor robot Caesar. This robot uses a special interaction method, mainly through voice or gestures for information interaction. This mobile robot is also the first mobile robot to be applied A* algorithm as a global path planning. The system, due to its special interactive mode, this robot is mainly used to serve the society, and it can communicate with some deaf-mute people at the same time [12]. After a while, a scholar named Breuer proposed a more powerful algorithm. This algorithm includes multiple aspects, including SLAM, human-computer interaction, etc. It is applied to mobile robots..

II. The Proposed Methodology

A. Smart Chip Embedded Design

To design an SOPC system, it is necessary to model the system functions and performance. At present, both software and hardware can be compiled through a dedicated description language. In the past, software and hardware were designed separately, and coordination problems were prone to occur. The software and hardware co-design makes up for the shortcomings; to achieve the software and hardware co-design, it must first be divided reasonably. For the SOPC system, it means that the resources occupied by the software and hardware on the FPGA can be balanced. After the division, it is necessary to carry out a unified description of them, including processing unit allocation, software and hardware division, and task scheduling; finally, software and hardware collaboration and simulation verification, feedback to the designer is software and hardware division, selection and scheduling, etc. aspect of choice.

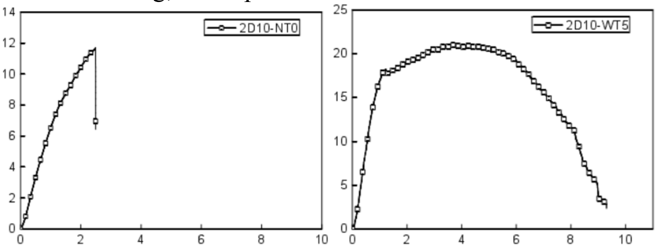

Fig. 1. Smart Chip Embedded Design

Multitasking provides a mechanism for responding to multiple, discrete events in the real world. The real-time kernel provides a basic multitasking environment. Multitasking constructs the illusion of multithreaded concurrent execution, but in fact the system kernel is interleaved execution according to a certain algorithm. Each task has its own context, that is, its own environment and system resources. When context switching, the context of the task is stored in the task control block. The context of the task includes the execution point of the task, that is, the registers and floating-point registers in the program counter of the task. Optional dynamic variables and function calls. A time slice timer core controls the value of the structure signal handle for debugging and performance monitoring. Traditional embedded system design starts with hardware and then software. The design cost is high and the cycle is long. When more and more functions are available. So the system is very complicated. As a result, the design and production time is too long, and the quality is greatly compromised, which cannot meet the needs of the market.

It can be seen from Figure 2.1 that the following shortcomings can be analyzed: the hardware and software are designed separately, the sense of hierarchy is low, the more system functions, the higher the complexity of the design, because the design process is completely manually operated by the designer, which requires a lot of time. At the same time, it is prone to errors; in the design, the hardware and software need to be tested repeatedly and modified continuously. The final product may not meet the designer's expectations and affect the effect; in the use of embedded processors, it is too single, it is not flexible and cannot be customized according to the requirements of the designer.

B. Embedded Real-Time Operating System

Due to the limited resources of the embedded real-time operating system, the system kernel is designed, the basic OS services are kept in the system kernel, and other functions of the system are moved out to the user space for implementation. The micro-kernel architecture design is shown in Figure 1. Microprocessors are generally related to interrupt and open interrupt instructions. The c language compiler used by the user must have a certain mechanism that can directly implement the interrupt-off/open-interrupt operation in C. Some C compilers allow inserting assembly language statements in the user's c source code. This makes it easy to insert microprocessor instructions to turn off/on interrupts. And some compilers turn off the interrupt from the C language, and open the interrupt in the extended part of the language.

$$x(k+1) = x(k) + v(k)\cdot\Delta T \tag{1}$$

$$y_k = f(net_k) \tag{2}$$

C/OS-11 defines two macros (macros) to turn off and on interrupts, in order to avoid different C compiler vendors from choosing different methods to handle off and on interrupts. uC/OS. The two macro calls in II are: OS_ENTER_CRITICAL() and OS_EXIT_CRITICAL(). Because the definition of these two macros depends on the microprocessor used, it is in the file OSCPU. The corresponding macro definition can be found in H. Each kind of microprocessor has its own 0SCPU. H file. The process of software and hardware co-design mainly includes the following aspects: System overall design planning: Make overall planning for the required functions of the system, that is, overall design and analysis of software and hardware, so as to determine the functions and related parameters. It can be described in HDL, SystemC, and natural programming languages, and can be modeled and simulated with Matlab, and finally verified on the FPGA chip, thereby effectively improving the design planning and speed of the system; the operating system that supports the operation of the embedded systemis called It is an embedded operating system.

The characteristic of hard real-time multitasking is the main feature of embedded system. If embedded system wants to improve the efficiency in its development process and shorten the tedious process in the development process of embedded system, it needs the existence of embedded operating system and embedded operation. The system is equivalent to the existence of the foundation when building a house, because of its existence, the portability of application software will increase. Embedded real-time operating system is a branch of the embedded operating system. The basic of system design is to ensure real-time performance. When developing an embedded real-time system, the first issue to consider is real-time performance, and secondly, how to improve the operating system. The most prominent feature of the embedded real-time operating system is that it can meet the time constraints and requirements of the application.

C. Embedded Real-Time System and Application Software Implementation

In the embedded real-time operating system, the sending and receiving of data can be realized by simple function calls, but because of the unpredictability of the message, it will increase the additional overhead of the system. Therefore, this article uses the embedded network communication protocol optimization method to solve the above-mentioned problems. For short messages, there is no need to implement too many functions in the system, so the level bypass is used to determine the protocol specification of the protocol level. According to the system function requirements, determine the necessary protocol level functions, combine the protocol implementation functions to be used into small modules, and at the same time remove the useless protocol levels to avoid the unnecessary overhead generated by the protocol level. Time management: timer interruption to achieve μC/OS- II time management, generally 10ms/time or 100ms/time, the designer designs the interrupt frequency according to the needs, and the interval time remains unchanged.

$$L = nE \tag{3}$$

$$Q_g = M(h_\alpha - h_E) \tag{4}$$

But only system functions related to this frequency can be called, because this is required by μC/OS-II. Such as: system time function. Task management: μC/OS-II is designed with ANSIC language, and has been ported to 8-bit to 64-bit CPUs. There are nearly 40 kinds of processors that have ported μC/OS-II. It can manage up to 64 tasks, of which 8 are reserved, and the remaining 56 can be applied by the designer. Since it gives each task its own independent stack address, it can switch between tasks at any time. And it allows the tasks with the highest priority to run first. □Memory management: Malloc and free functions are used in ANSIC to dynamically allocate and release memory.

The shortcomings are: memory fragmentation is prone to occur after repeated use; memory management algorithms lead to uncertain time. If the system wants to run smoothly, the operation between various tasks must have the ability to coordinate with each other, and good communication between tasks is an important basis for embedded systems to synchronize tasks and tasks. μC/OS-II can use semaphore, message queue and message mailbox to communicate between tasks. Users who want to use these three methods for communication and data sharing between tasks need to use task control blocks. The event control block in μC/OS-II is also a very important module, which contains a list of event waiting.

III. EXPERIMENT

The layout design of the smart chip is shown in the figure.

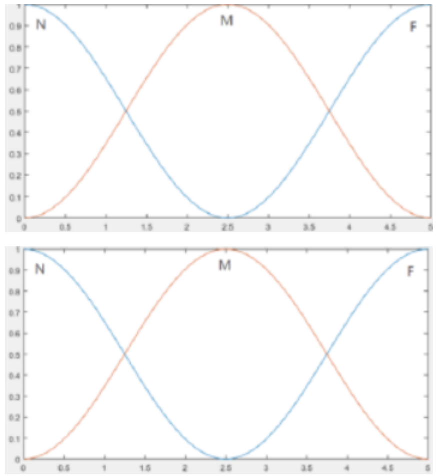

Fig. 2. Layout design of smart chip

The embedded real-time operating system is shown in the figure.

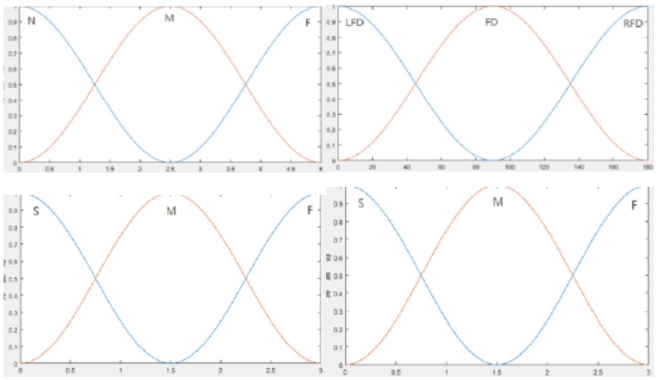

Fig. 3. Embedded real-time operating system

Embedded real-time application software is shown in the figure.

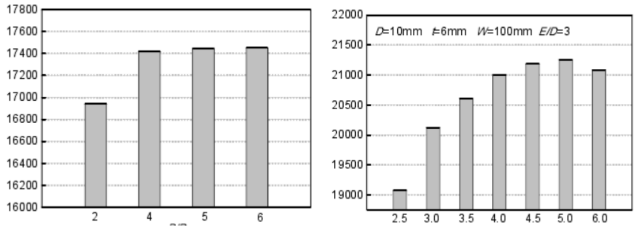

Fig. 4. Embedded real-time application software

IV. CONCLUSION

This paper focuses on the research and analysis of embedded real-time operating system. Based on the original research data, designs an embedded real-time operating system based on computer network technology, and uses computer network technology to further optimize the internal structure and functions of the system, highlighting the embedded system the advantages. After the system design was completed, the actual application level of the designed real-time operating system was proved through comparative experiments, and the purpose of extending the service life of the system CPU was achieved. However, in the design process, there are still some issues worthy of discussion and excavation.

REFERENCES

[1] Lu Shun. Development of embedded real-time operating system in single-chip microcomputer application program[J]. Electronic World, 2019(3): 2.

[2] Chen Tong. Program design and implementation of embedded operating system based on STM32[J]. Computer Fan, 2018, 000(018):80.

[3] Chen Miao. Design and implementation of a new intelligent interactive terminal based on μC/OS-Ⅱ[D]. Hunan University, 2018.

[4] Sun Xiao. Design and implementation of embedded operating system DeltaSVM on ARM architecture [D]. University of Electronic Science and Technology of China, 2018.

[5] Wu Hao. Design and implementation of intelligent music making real-time perception monitoring system based on industrial interconnection [D]. University of Electronic Science and Technology of China, 2019.

[6] Sun Xiao. Design and implementation of embedded operating system DeltaSVM on ARM architecture [D]. University of Electronic Science and Technology of China, 2019.

[7] Zheng Wei, Sun Xufei, Xie Rong. Design of Residential Trash Can Monitoring System Based on LoRa Technology[J]. Microcontrollers and Embedded System Applications, 2020, 20(3): 4.

[8] Ou Linlin, Zhang Qiang, Yu Xinyi, etc. Intelligent fitness cycling platform system based on FreeRTOS embedded real-time operating system:, CN109100968A[P]. 2018.

[9] Chen Yubing. Development of farmland environment monitoring and irrigation control system based on Internet of Things [D]. Xi'an University of Posts and Telecommunications, 2019.

[10] Hou Peimin, Shao Feng, Chen Xiaojun, et al. A smart meter based on embedded operating system:, CN111537791A[P]. 2020.

[11] Chen Liqi. Design of embedded multi-motor intelligent control system based on single-chip microcomputer[J]. Computer Measurement and Control, 2020, 28(5):6.

[12] Mao Demei, Wang Mingzhu, Wang Benyou, et al. Research and design of embedded intelligent high-voltage circuit breaker condition monitoring system[J]. 2021(2013-2):47-50.

[13] Wu Zhenlei, Gu Chuchu. Smart pill box design based on stm32 embedded technology [J]. Gansu Science and Technology, 2020, 49(4): 3.

[14] Lin Yuewei, Mou Sen, Jiang Haijun. Technical practice of smart car based on ARM and embedded Linux[J]. Computer System Applications, 2018, 27(08):245-250.

[15] Shi Yuanbo. Design of embedded real-time data acquisition and control system[J]. 2021.

[16] Zhang Di, Dong Fei, Gao Bin, et al. Design of smart miner's lamp based on embedded real-time operating system[J]. Industry and Mine Automation, 2018(2): 5.

[17] Shan Wensong. Design and implementation of a single-phase power quality monitor based on STM32 [D]. Shandong University, 2018.

[18] Hu Guochen. Research on uC/OS-II embedded real-time operating system based on ARM[J]. Information Recording Materials, 2018, 19(12):3.

[19] Li Qian, Liu Sen, Liu Maosheng, et al. BMS software design based on embedded operating system uC/OS-Ⅱ[J]. Electrical Application, 2020, 39(2): 6.

[20] Liu Meihua. Research on Intelligent Control System of High Speed Arc Discharge Machining Machine [D]. 2019.

[21] Chen Ying. Design and implementation of low-power face recognition access control system [D]. Hunan University, 2018.

[22] Liu Jing. Research on Gas Flow Control System of Atomic Fluorescence Analyzer[D]. Chengdu University of Technology, 2019.

[23] Sun Wenxue. Design and implementation of home electricity intelligent monitoring system based on STM32 [D]. Foshan University of Science and Technology, 2020.

[24] Wang Liang. Design and implementation of motion control system for small unmanned surface vessel [D]. Changsha University of Science and Technology, 2018.