Implementation of the No Packet Loss Network Communication in VxWorks

Linfang Zhang

Beijing Naval Representative Office of Armaments and Accessories, Beijing 100082, China

Abstract

By VxWorks, the real-time control system of certain equipment adopts the UDP/IP network communication, which requires that the control process has strong real-time character and complex task scheduling. In the work condition with frequently interactive network tasks, the control system is easy to lose packets, and to solve this problem, this article adopts the double-loop buffer structure processing mechanism to realize the no packet loss network communication method, which could largely enhance the reliability of the network communication and improve the stability of the control system. Based on the excellent multi-task scheduling ability of VxWorks, this method could essentially enhance the efficiency and reliability of software. By testing, this method is efficient and feasible.

Keywords: Vxworks, UDP, Network communication, Real-time control system

1. Introduction

VxWorks operation system is an embedded real-time operation system (RTOS) designed and developed by US WindRiver Company in 1983. With the core of high performance and friendly user development environment, it has been one of the most excellent systems in the embedded real-time operation systems. By its good reliability and excellent real-time, it has been widely applied in the high-grade, high-precision, and advanced technologies such as communication, military affair, aviation, and spaceflight, and the domains with higher requirement of real-time such as satellite communication, military maneuver, ballistic trajectory control and guide, and plane navigation. It has even been used in F-16, FA-18, B2 stealth bomber, and patriot missile, and the Mars detector which had landed on the surface of Mars in April of 1997.

Based on the analysis of VxWorks multi-task mechanism and task scheduling, by comprehensively considering the receiving and processing of the large-scale data for the practical control system, the project adopting the double-loop buffer structure and the priority setting is proposed in this article. When the computer could receive the quality of data, this project could solve the problem of receiving and processing large-scale data in emergencies in essential. Many sub-systems and control systems exist in a complete battle system, and this project could reliably complete the running of the battle system.

2. Analysis of the characteristics of VxWorks

2.1 Real-time

The character of real-time means the ability that could exert regulated functions in the limited time and response the exterior asynchronous events. The real-time is measured by the time that completes regulated functions and makes response.

The character of real-time of VxWorks is very good, and the expense of the system is very little, and many public programs such as course scheduling, course communication and interruption processing are very effective, and they often induce short delays. The preemptive priority scheduling and round-robin scheduling of the task control in the multi-task mechanism provided by VxWorks could sufficiently ensure the reliable character of real-time, and it makes same hardware collocation could fulfill strong requirement of real-time, and level bigger space for the development of application.

The implementation of the multi-task system is completed by the conversion and control among multiple tasks by CPU. In the multi-task system, CPU could exert only one task in one period, and it will transfer the access of CPU to next task after the task is executed. The modern real-time operation system is the concept based on multi-task and task communication, and one multi-task environment allows the real-time application to construct the set composed by multiple independent tasks, and each task is executed independently, and possesses its own complete computer resource. The running of multi-task could largely exert the utilization of CPU, and realize the largest modularization of the application. In the real-time system application, the largest character of the multi-task is that developers could divide complex application, enhance the parallelization and modularization of software.

Based on the task scheduling mechanism of PRI, the VxWorks multi-task core Wind uses the interruption driver, and has quicker context switch time and lower interruption delay. VxWorks task could interview most system resource directly or by the sharing way, and have separate context to maintain respective control courses. Each task has its own context (including the CPU environment and system resources). The task is run by the core scheduling of the system, and in the context switch, the context of the task is saved in the task control block (TCB).

2.2 Task scheduling of VxWorks

VxWorks has two kinds of task scheduling strategies such as the preemptive priority scheduling and the round-robin scheduling. The system has 256 PRIs. The preemptive priority scheduling means that when the task with lower PRI is being executed, another task with higher PRI would enter into the preparation and occupy the resource of CPU, and execute the task with higher PRI until the task with higher PRI releases CPU (completing, hanging, and blocking), and the interrupted task with lower PRI could continue to the executed. The preemptive priority scheduling could ensure the quick response of the real-time system, but it also has many problems. For example, when several tasks with same PRI exist in the system, the single task will occupy the CPU alone until it is executed. And if this task occupies too much time, it will influence the running of other tasks with same PRI. When the round-robin scheduling is used, the system could allocate same one period for the tasks with same PRI, and these tasks will occupy the CPU by the round-robin way, and the tasks with different PRIs could be scheduled by the preemptive priority scheduling mode.

2.3 Inter-task communications

Multiple communication forms exist in tasks, and the kernel supports various UNIX measures such as semaphore, message queue, pipe, signal, and socket. The semaphore is mainly used to protect the critical resources by the mutex way or complete the interruption and synchronization among tasks. The message queue is mainly used in the message drive mechanism. The pipe is the communication mode to transfer data by the first-in-first-out mode among tasks based on the file system. The signal equals to the soft interruption which is used to display the false information. The socket is mainly used in the network communication. When large-scale data need to be exchanged among tasks, it needs not special measure because Vxworks could operate physical memory to directly interview the whole storage region, which is different with the sharing storage application in Linux and the storage mapping file mode in Windows.

2.4 Network support of VxWorks

VxWorks supports two network drivers. The first one is the BSD which supports general BSD4.4 network and API, and is similar with most BSD networks. The other one is the END which is special in VxWorks, and according to the interface programming of VxWorks MUX, END should be converted to the form of BSD on the bottom layer.

3. Characteristics of the UDP receiving method based on the loop buffer queue

3.1 No packet loss of data receiving

In the communication, if the message receiving frequency is too high, some messages will be lost and need to be received because the message is processed too late or the information is lost. The loop buffer could divide the read-in data and the read-out data, and because the PRI of the task that the data receives is the lowest, so when other programs quit CPU, this task will continually inquire the end for whether the data receive, and once the data receives, the task will send the data to the buffer region, which could the data only respond the data receiving on the bottom layer, and ensure the data could be received quickly and will not be lost, and the intention that the no packet loss of data receiving could be achieved.

3.2 Advantages of no packet loss

The loop buffer could realize the continual processing of information, reduce the information processing time, and realize the communication mechanism of no packet loss. This communication mechanism also has following advantages.

3.2.1 High-speed polling data end

In the communication mechanism proposed in this article, the UDP is on the lowest layer of the system task when it receives the task, and other tasks are the tasks which need be executed in high speed, i.e. the time occupying CUP is very short, so much time could be given to UDP to receive the task which could occupy CPU in a long time and be in the execution state, and the UDP data receiving end could realize the round-robin scheduling.

3.2.2 Enhancing the work efficiency of the processor

In the communication mechanism of this article, without man-made task delay and special TASKDELAY () Function, the receiving, explaining and processing of information could be executed by the scheduling program according to the PRI, so CUP will not be delayed specially, and the working efficiency of CPU will be enhanced.

4. UDP/IP network communication mode based on the loop buffer structure

4.1 Introduction of UDP/IP network communication

UDP is called as the user datagram protocol, and it is the protocol facing the unconnected message communication management mode, and it is used to exchange the data package in the connected network environment. In ISP, UDP belongs to the transportation layer protocol, and the implementation of its function is based on the data link layer, the physical layer, and the network layer. As viewed from the protocol, UDP is on the IP protocol (the network layer), and UDP is transported in the network by IP protocol. UDP is another method of IP, and like IP, UDP uses the IP protocol to acquire the data unit (datagram), and being different with TCP communication, UDP doesn't provide the grouping and reassembly service of package (datagram), and UDP will not rank datagram, that means the program must confirm whether the information could completely and correctly arrive to the destination. If the transportation speed of the data package in the network wants to be further enhanced, the efficiency of UDP is much better than TCP. UDP contains two services which don't be provided by IP layer, and it could use the appointed end number to distinguish different users' requests, and provide the parity check. In the mode of OSI, UDP is in the further layer like TCP, i.e. the transportation layer.

4.2 UDP loop buffer structure communication mode

UDP/IP loop buffer communication structure is seen in Figure 1, and the whole system includes four layers.

(1) UDP data receiving task

When multiple tasks run in parallel, the receiving of message should be considered first, and the PRI receiving this task is the lowest one, i.e. the system could be in the message receiving state in most time. When other tasks of the system quit CPU, this task will be executed repeatedly, and the system will continually inquire the state that the network receives data, and once the data are received, the data will be transferred to the buffer region, and the message is begun to be processed.

(2) Message explanation

The PRI of this task is higher than the task that UDP data receives task. This task is to explain and analyze the data received from the network. When other tasks are in leisure, this task acquires data from the loop buffer region, and springs the execution of this task to process the data.

(3) Message processing

According to the parameters in the message analysis and task processing and the parameters of the system, the program will enter into different processing flows. This layer contains many processing types such as data display, data computation, and message sending. The execution of data type may be a type, or multiple types, so in the message processing, multiple tasks will be established, and the same PRI will be set up, and the task will be executed by the time round-robin way.

(4) Data transmitting

The PRI of data transmitting is the highest one in this project, i.e. once the message comes into being, it will be transmitted at once. This task is to complete the write-operation of data caching, so this task could be completed in the shortest time.

4.3 Task synchronization at the data receiving end

The synchronization among tasks adopts the counting semaphore and message queue, and the operation of the semaphore and message queue are among tasks is seen in Figure 2.

The software will cut a loop buffer region for UDP to receive task, and create the control semaphore, and implement the search management in the buffer region, and its initial value is 0. When the information receives the buffer semaphore, and the information receives the task and transfers the SemGive () and adds 1, and the message analysis task will transfer SemTake () and reduce 1, and the value of semaphore represents the current information quantity in the queue. Because the information receiving loop buffer is operated by same one task, it must protect the data. And to ensure the correctness of the buffer region data operation, one mutex semaphore is used to implement the write-read protection for the loop buffer region.

For the information transmitting semaphore and the message processing task releasing semaphore, because the time that the data transmitting occupies too less time of CPU, the PRI of this task is the highest one, and once the semaphore is acquired, it will scheduled and transmitted at once.

4.4 Confirmation of task priority

In the system described in this article, the scheduling of PRI is adopted to execute the task with high PRI, and the using right of CPU could be acquired. At the same time, the mutex semaphore is used to ensure the exclusive right of the buffer region, and before it ends, other tasks could not operate the buffer region, so the correctness and uniqueness of the operation could be guaranteed. For the interruption service program, its PRI is higher than other tasks, and here, other tasks which are being executing will be hung without condition, so the buffer region could not be operated in the interruption service program. The practical method is that the interruption service program uses the mode of message queue to add the interruption information in the task queue, and the latter will receive normal task scheduling, and acquire CPU in proper condition and complete the operation in the buffer region. When all tasks are blocked, CPU will be in leisure. The PRI setting of the tasks at the controlled end is seen in Table 1.

5. Conclusions

The UDP network communication based on loop buffer in this article could fully utilize the multi-task support of VxWorks, adopt the semaphore and message queue to ensure the synchronization and communication among tasks, and use the task PRI to ensure the memory protection. The working method of the loop buffer queue could make the network communication without packet loss, and have many characteristics such as timely response and high software efficiency, and it could widely applied in the future.

References

Kalmar. (2005). *System Structure Programming and Design of the Embedded System (Gravure).* Beijing: Tsinghua University Press. Feb of 2005.

Wind River Company. (2004). *Tornado Users Guidance/ VxWorks Developer Guidance Series.* Beijing: Tsinghua University Press. Oct of 2004.

Wind River Company. (2003). *VxWorks Network Programmer Guidance.* Beijing: Tsinghua University Press. Sep of 2003.

Table 1. Task priority setting and CUP occupation table

No.	Task	Priority	Execution condition	Pending condition
1	UDP message receiving task	Lowest	Repeatedly inquiring ends without others tasks	Once receive data and treat data
2	Message parsing task	Lower	Message receiving task and releasing semaphore	Message parsing ends
3	Information execution task	Higher	Triggering information processing	Information processing ends
4	Message sending task	Highest	When messages are needed to be sent	Sending ends

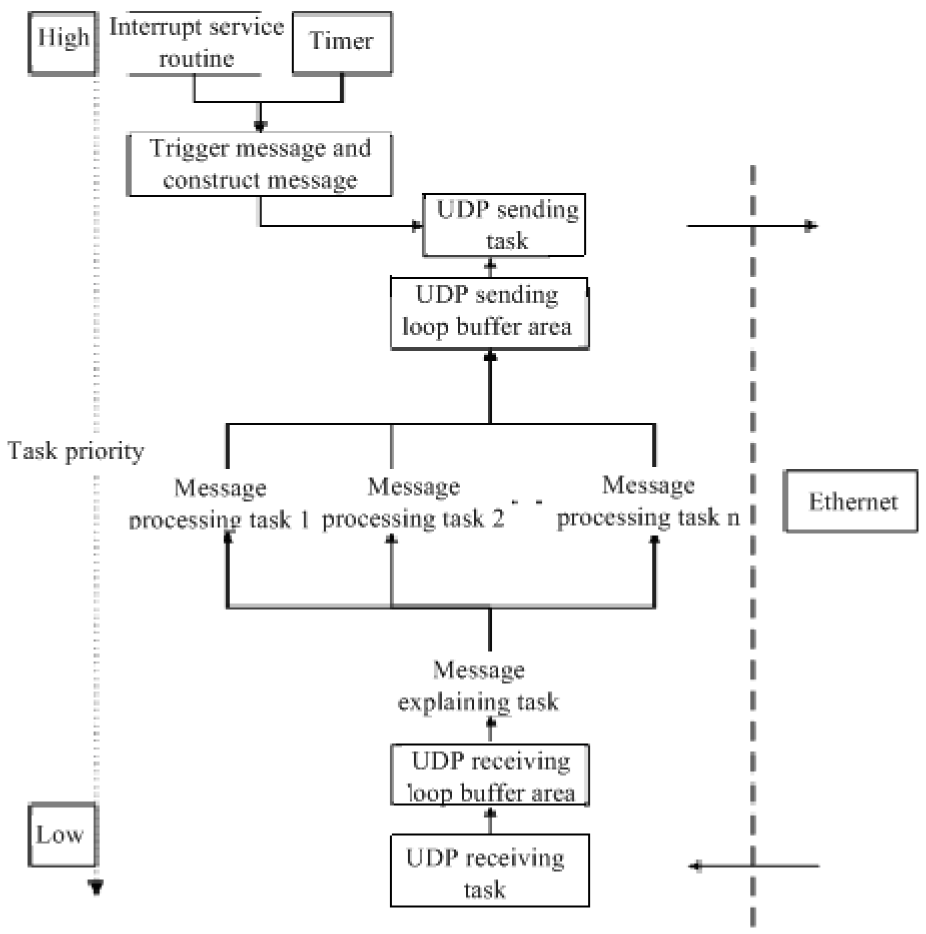
High
Interrupt service routine
Timer
Trigger message and construct message
UDP sending task
UDP sending loop buffer area
Task priority
Message processing task 1
Message processing task 2
Message processing task n
Ethernet
Message explaining task
UDP receiving loop buffer area
UDP receiving task
Low

Figure 1. UDP Loop Buffer Communication Structure

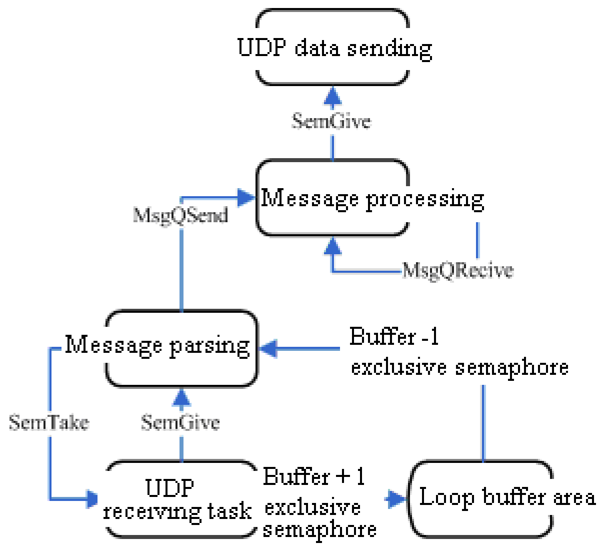

Figure 2. The Operation of Semaphore and Message Queue inter-task