Advanced Science and Technology Letters
Vol.31 (MulGraB 2013), pp.158-162
http://dx.doi.org/10.14257/astl.2013.31.35

Hardware scheduler of Real-time Operating System

Xue Liu, Yun-feng Ding,Guo-hua Zhu,Yan Li

Harbin University of Science and Technology, China
Liuxue1990@gmail.com

Abstract. In this paper, new scheduling algorithms of μC/OS-II real-time operating systems and their hardware implementation are shown. The new scheduling algorithm breakthroughs the limit of the number of total task, and hardware task scheduler was used to select tasks from ready list instead of the original ready table, maintaining the correctness of the software scheduler and improving efficiency of the entire real-time system. This algorithm redesigns in VHDL hardware description language, simulated in ISE 8.2 and implements in FPGA. The new task scheduling algorithm and hardware scheduler decrease the cost of hardware logic and the spending of running time.

Keywords: Task management; Lists of ready tasks; Hardware Implementation; μC/OS-II; FPGA;

1 Introduction

With the development of embedded technology, RTOS (Real-time Operating System) has been increasingly applied in areas of embedded system [1]. To the existing real-time operating systems implemented by software, it is hard to continually enhance their real-time performance by simply improving their scheduling algorithms. So, the combination of improvement of algorithm and implementation of hardware logic to realize task management and task scheduler becomes necessary. Hardware logic is independent from CPU and would not cost CPU resource [2], so this method has the value of research and practice.

2 Data structure of Task Control Blocks

The core technology for a real-time operating system is its task scheduling mechanism. In order to restore the states and other information of task running, after the establishment of every single task, a task control block (TCB) should be created to manage and schedule this task. The TCB data structure used by μC/OS-II is to maintain the state of a task when it is preempted [3].

So, the crux of the matter is to redesign the whole data structure of TCB to not only be compatible to μC/OS-II TCB but suitable for implementing in the registers inside FPGA. This data structure should also be easy for comparing the priorities of tasks and then sorting the task priority. The date structure can be designed as listed below.

ISSN: 2287-1233 ASTL

OSTCBId is the register of task IDs in the operating system to mark every task. OSTCBStat is the register of current state of the task, reading or writing which can return or change the current state. OSTCBPrio is the register of priority of the task. OSTCBDly is the register of delay time, used when the task needs to be delayed for a certain number of clock ticks. OSTCBStkSize is the registers of the size of the stack. OSTCBStkBottom is a pointer to the task's stack bottom. This data structure keeps major items in TCB of μC/OS-II in order to be compatible with the original μC/OS-II TCB. Some items are not implemented in FPGA register because it will not improve the real-time performance of the system.

3 Algorithm of the scheduler

Task management and scheduling are the essential function of a real-time operating system. Scheduler should determine which is the next task that needs to execute and operate the stacks to perform the context switch [4]. It needs to maintain a group of task lists. Every times when meeting the requirement of scheduling, scheduler sorts all ready-to-run tasks based on priority of the task and then selects the task with the highest priority. Algorithm used in this paper modifies the original μC/OS-II algorithm. The algorithm adds comparison of the same priority level. Thus, if the number of OSTCBPrio is the same, the task with check the number of the list of OSTCBPrio, which guarantee scheduler always choses the task with the highest priority and waiting the longest time in the list.

The priority is corresponding to the task, so TCB can be determined when the scheduler selects the highest priority task in the ready list. This kind of function is represented in the OSTCBPrioTbl[], so when certain priority has been determined, scheduler can index the TCB data structure from OSTCBPrioTbl[] table. While if there one priority level links multiple tasks, then instead of pointing to one single TCB data structure, the item in OSTCBPrioTbl[] points to a list of TCB data structures in which TCB data structure has the same priority levels. The connection of OSTCBPrioTbl[] and TCB items shows in Fig. 1. Examples of OSTCBPrioTbl data structure.

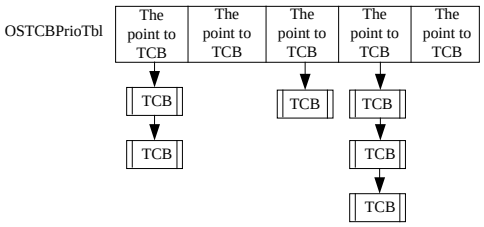

Fig. 1. Examples of OSTCBPrioTbl data structure

In order to implement this algorithm in a way of high efficiency, combinational logical circuits have been adopted to create this hardware task scheduler, showing in Fig. 2.. This task scheduler sort tasks according to their priorities. It directly links to the state registers of all tasks, as long as the scheduler detects one that changes, it will

cause a reschedule of all tasks. The data selectors directly connect with TCB registers. They, based on the priority level, output OSTCBStat to corresponding output bit in REG_X registers, and then all REG_X registers perform logical OR, and send the results to PRI_REG registers.

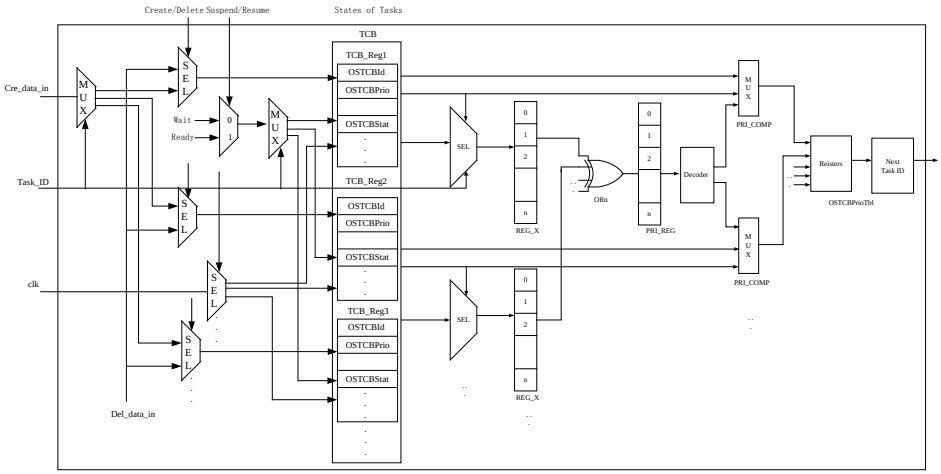

Fig. 2. Hardware circuits of the task scheduler

4 The results and analysis

In order to test the correctness and high efficiency of the scheduling algorithm shown in the paper, the whole system has been implemented by VHDL hardware description language simulated in ISE 8.2 design suite. The functional simulation of hardware task management and scheduler shows in Fig. 4.

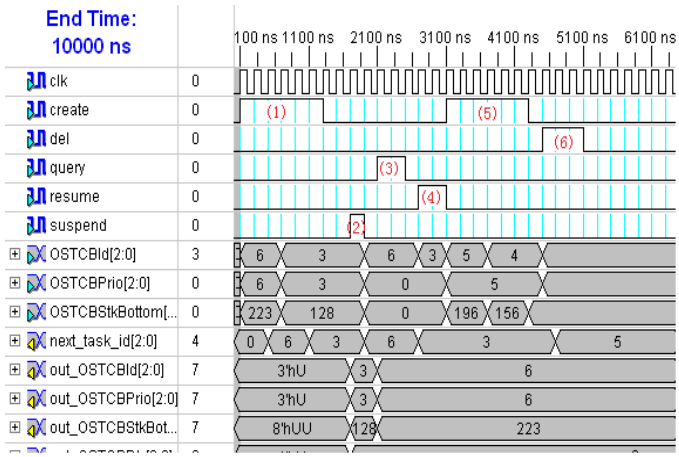

Fig. 4. The simulating graph of task management and scheduler

Suppose there are 8 tasks running in the operating system, the usage of hardware resources in entire system show in Table 1. The practical usage of hardware in XC2VP -30FF896C platform. According to Table 1, resources of FPGA and performance meet the need of real-time operating system.

Table 1. The practical usage of hardware in XC2VP30FF896C platform

Logic Utilization	Used	Available	Utilization
Number of Slices	491	13696	3%
Number of Slices Flip Flops	712	27392	2%
Number of bonded IOBs	81	556	14%
Number of 4 input LUTs	350	27392	1%
Number of GCLKs	8	16	50%

The analysis of the results verifies the correctness and high efficiency of using hardware logic to implement operating system scheduler, which meets the requirement of real-time operating system.

5 Conclusion

This paper mainly redesigns the scheduler to meet the limit of hardware logic as well as high efficiency and improve the algorithm by adding ready lists, which will support tasks with the same priority numbers and break through the limit of the number of total task. This paper use registers inside FPGA to implement the redesigned TCB data

structure and directly links it to the scheduler, which maximizing the potential performance of multiple tasks. So, hardware scheduler has its value of research and real life.

Acknowledgement. This study was supported by the National Natural Science Foundation of China (No. 61103149), the Education Department Foundation of China (No. 12521100), the Technology Innovation Talent Research Foundation of Harbin (No. 2013RFXXJ034) and Innovative and experimental program for college students of china.

References

1. T.Nakano, U.Andy, M.Itabashi, A.Shiomi and M.Imai: Hardware Implementation of a Real-time Operating System. Proceedings of the Twelfth TRON Project International Symposium, IEEE Computer Society Press, pp. 34-42(1995)
2. M. Vetromille, L.Ost, C.A.M. Marcon, C. Reif, F. Hessel.: RTOS Scheduler Implementation in Hardware and Software for Real Time Application. Rapid System Prototyping, 2006. Seventeenth IEEE International Workshop, 163 - 168 Digital Object Identifier 10.1109/RSP.2006.34.(2006)
3. Jean J. Labrosse: MicroC/OS-Ⅱ The Real-Time Kernel. Translated by Shao Beibei and so on. Beijing: Beijing University Press:178-185(2001)
4. Cui Jianhua, Sun Hongsheng, Wang Baojin. The design and realization of hardware real-time operating system. The of application computer technology,34-37(2008)

