2014 Sixth International Conference on Measuring Technology and Mechatronics Automation

Research on Multi-core Heterogeneous Operating System Architecture

ZhouRong

Sichuan Vocational & Technical College, Suining, Sichuan 629000, China

Zicdma@163.com

***Abstract*—According to the researches on multi-core processor software framework, on basis of the hierarchical software framework including data plane, control plane and management plane, this paper proposes a software architecture for heterogeneous operating system. It takes consideration of the developmental difficulty of management plane, the generality of control plane and the processing efficiency of data plane. We discuss the multi-core heterogeneous architecture in aspects of data plane programming model, heterogeneous operating system signal interaction and data interaction respectively. Experiments show that this multi-core based heterogeneous operating system architecture can deliver better network protocol processing performance than the general architecture.**

Keywords-Multi-core processor; Operating System; Heterogeneous system; Software Architecture

I. INTRODUCTION

Multi-core network processors can provide high speed parallel processing and specific network acceleration, but the flexibility in programming is still a problem so far. Because of the particularity and complexity of the architecture, programmers must directly handle a variety of hardware resources, such as processor cores, memory, communication mechanism, etc. But in the traditional software framework, it is managed by the operating system (or OS for short). Due to the hardware details, the developing efficiency on multicore processors is going much lower than ASIC.

According to these problems, literature [1] pointed out that designing a software framework based on multi-core network processor should consider the following aspects. First, the software framework must abstract out the hardware architecture details for the programmers. Second, according to different network applications, the programming environment should automatically allocate and manage the hardware resources. Third, the software framework should support dynamic resource allocating mechanism, which can provide the performance guarantee for network processing systems. Fourth, the software framework should be redefined and extensible, as well as portable for applications.

According to researches on multi-core processor software framework, on basis of the hierarchical software framework including data plane, control plane and management plane, this paper proposes a software architecture for heterogeneous operating system. Specifically, it divides the multi-core network processor cores into two groups. One group runs general operating systems, such as Linux; The other runs network processing systems [2][3][4], for example, the real-time operating system, the embedded execution environment, and some specific accelerating execution environment, such as Cavium SE execution environment.

II. ARCHITECTURE

With the improving of the people's life, the request to the sanitary implement is becoming higher and higher. At the same time, it is requested that: (1) full fill the essential function, (2) stress the shape beauty, and (3) saving water. The quality of the sanitary implement has been requested to include exact size tolerance, fine external appearance, physics function, flushing function etc. Flushing function is the ability that the bedpan ejects filth, washes it clearly while controlling the amount of water. Without question, flush function is one of the uppermost functions. According to the flushing way, bedpan can be divided into wash-down type, siphonic type and ejecting siphonic type. Wash-down bedpan rush out filth by the impact of the water, the power is decided by potential energy of the water in tank subtracts losing energy. Siphonic bedpan's flush power primarily relies on siphonic power.

Given the characteristics of multi-core network processors, we consider software architecture for heterogeneous operating system should focus on the following aspects.

(1) The developmental difficulty of management plane. Because of involving many general management protocols and human-machine interactive functions, the software logic maybe quite complex. To reduce the developmental difficulty, we should try to exploit the general OS programming interfaces and some portable software.

(2) The generality of control plane. Because of interacting with data plane directly, it should try to adapt different hardware platforms via exploit the general and specific OS interfaces.

(3) The processing efficiency of data plane. As the key part of this architecture, data plane need to deliver high performance via reasonable organization, mapping and scheduling. And it can achieve a linear speed processing by design an efficient data plane model.

978-1-4799-3435-5/14 $31.00 © 2014 IEEE
DOI 10.1109/ICMTMA.2014.137

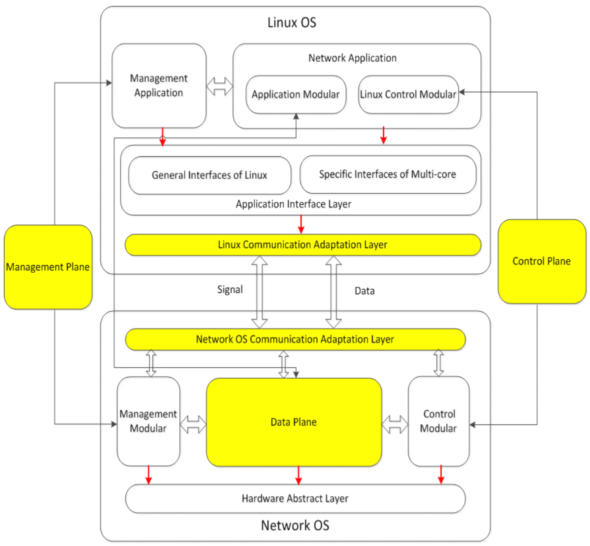

Figure 1. Heterogeneous operating system architecture based on multi-core processors

Based on these analysis, the multi-core heterogeneous OS architecture in this paper is shown as Figure 1. In this architecture:

(1) The management plane includes human-machine interactive functions on Linux and management modular on network OS;

(2) The control plane includes the network applications on Linux and control modular on network OS to control the running of data plane;

(3) The data plane processes the network packets efficiently via handling the hardware resources such as processing cores, memory and co-processors directly;

(4) The application interface layer provides general interfaces of Linux such as socket API and specific interfaces of multi-core processors to programmers to develop network and management applications;

(5) The communication adaptation layer provides interaction between heterogeneous operating systems through inter-core communication for signal and shared memory for data;

(6) The hardware abstract layer provide convenient hardware access interfaces to developers such as inter-core communication interfaces, memory management interfaces, co-processor programming interfaces, etc.

III. Data Plane Model

Data plane is the key part of the entire system. The programming model of data plane should be efficient in developing and processing by reasonable organization, mapping and scheduling, which is crucial to the network system performance. According to the tasks implementation on processing cores, the multi-core topological structures can be divided into three types as following [5].

(1) RTC (Run to Complete) model. Each core can handle all of the network processing subtasks independently and concurrently.

(2) S-SPL (Single Software Pipeline) model. The processing cores work together as a pipeline. Each core in different level can accomplish a specific network processing subtask.

(3) P-SPL (Parallel Software Pipeline) model. By integrating RTC and S-SPL, P-SPL expand the core numbers of each processing stages in S-SPL. P-SPL can effectively enhance the processing capacity and eliminate the bottlenecks in pipeline structure.

Researches [6][7] indicate P-SPL model can take account of processing performance and expandability in network processing and can exploit the parallelism advantages of multi-core processor. In this paper, we design the data plane model in P-SPL to take advantage of multi-core processor and ensure software modularization and expandability. Figure 3-1 illustrates an example of the heterogeneous OS data plane in P-SPL model.

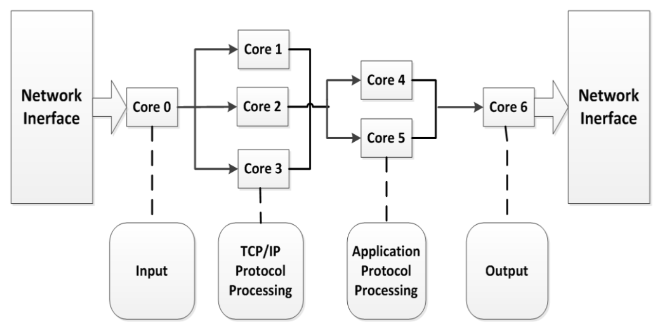

Figure 2. P-SPL data plane model

IV. Communication Adaptation Model

A. Signal Communication

In general network processors, inter-core communication should be handled by interrupts. The interrupts can bring about a lot of context switch and lead to inefficiency. In multi-core processors, a specific hardware unit is integrated on the chips to assist inter-core communication and synchronization.

In our system, the processing cores can be divided into several groups. We define a message structure named Work to express the communicating signals in hardware view. The cores in the same group can receive and process the Works with the same ID as the group. Therefore, when cores communicating, it should divide them into different groups and send message in Works with the same ID as the target core group ID. The signal communication mechanism with Work can reach higher efficiency than interrupt. The Work transmitting and receiving overhead is 60-120 cycles approximately, while an interrupt will take 200-300 cycles.

B. Data Communication

When heterogeneous operating systems running on multi-core processors, the reasonable memory allocation and sharing should be considered carefully. In our system, heterogeneous OS including Linux and Network OS are implemented on one or several cores of the multi-core processor respectively. Therefore, we logically divide the physical memory of the multi-core platform into three parts: exclusive memory for Linux, exclusive memory for Network OS and shared memory for heterogeneous OS communication. The shared memory mechanism provides the data interacting between heterogeneous OS. The implementation in detail is described in the previous section.

V. EXPERIMENTAL RESULTS

Based on the multi-core platform Cavium CN5860, without loss of generality, we choose to test and contrast the TCP/IP performance of the heterogeneous OS architecture in this paper and standard Linux OS. In order to eliminate the limitation of hardware equipment performance, we test TCP/IP performance in the case of different number of gigabit network interface cards (or NIC for short).

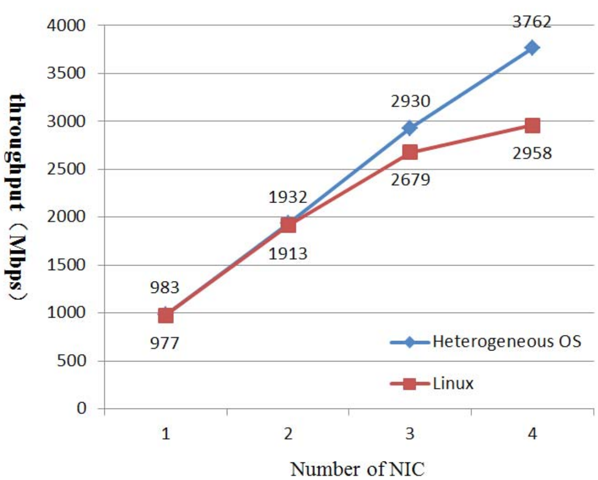

Figure 3. TCP/IP data throughput of heterogeneous OS and Linux

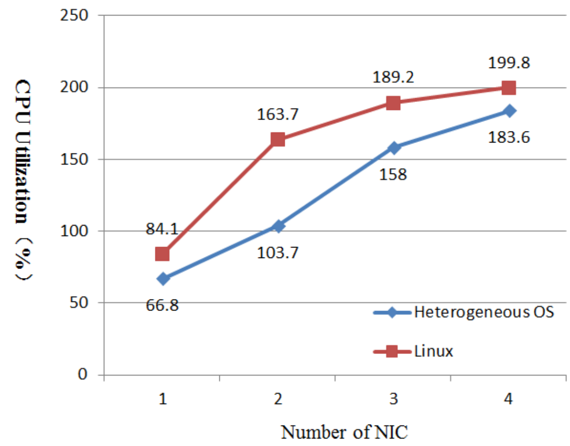

Figure 4. CPU utilization of heterogeneous OS and Linux

As is shown by figure 3, when testing on one or two NIC, TCP/IP data throughput performance of heterogeneous OS and Linux keep almost the same. The reason can be seen from figure 4, both of the heterogeneous OS and Linux are not reaching the limit of CPU utilization. And that's the reason why the performance can approach the data throughput limit of gigabit network interface. But it also can be seen that the heterogeneous OS consumes less CPU resource than Linux.

In tests on three NIC, heterogeneous OS can deliver TCP/IP data throughput of 251Mbit/s higher than Linux, while CPU utilization 61.6% fewer than the latter. In tests on free NIC, heterogeneous OS can deliver TCP/IP data throughput of 804Mbit/s higher than Linux, while CPU utilization 53.2% fewer than the latter. As is shown by figure 3, when testing on three or four NIC, TCP/IP data throughput performance of heterogeneous OS is significantly high than Linux. The reason can be seen from figure 4, heterogeneous OS can consume less CPU resources than Linux. When CPU utilization becomes the bottleneck, the advantage of heterogeneous OS can enhance the protocol processing performance evidently.

VI. CONCLUSIONS

When running general operating system on multi-core processors, such as Linux, all of the network protocol processing is handled by the OS kernel, which brings lots of interrupts and process scheduling overheads. Therefore, the processing efficiency and CPU consumption of general OS on multi-core is always far from satisfactory.

According to the researches on multi-core processor software framework, on basis of the hierarchical software framework including data plane, control plane and management plane, this paper proposes a software architecture for heterogeneous operating system. It takes consideration of the developmental difficulty of management plane, the generality of control plane and the processing efficiency of data plane. We discuss the multi-core heterogeneous architecture in aspects of data plane programming model, heterogeneous operating system signal interaction and data interaction respectively. Experiments show that this multi-core based heterogeneous operating system architecture not only can deliver better network protocol processing performance than the general architecture, but also consume less CPU resources.

REFERENCES

[1] JIANG HanPing. Research and Implement on Software Framework for Multicore Network Processors [D]. Wuhan University of Technology, 2008.

[2] Ghernaouti-Hélie S. Network Operating System[J]. 1998.

[3] Gude N, Koponen T, Pettit J, et al. NOX: towards an operating system for networks[J]. ACM SIGCOMM Computer Communication Review, 2008,38(3):105-110.

[4] Rashid R F, Robertson G G. Accent: A communication oriented network operating system kernel[M]. ACM, 1981.

[5] HE Pengcheng, et al. Packet scheduling algorithm on multi-core network processor towards streaming media [J]. Application Research on Computers, 2010, 27(10): 3911-3914.

[6] HE Peng-cheng, et al. On Architecture of Multi-core Packet Processing System [J]. Microcomputer Applications, 2010(009):12-20.

[7] GUO Xiu-yan, et al. Multicore Frame of Sending Real-time Streaming Data for Video-on-demand System [J]. JOURNAL OF CHINESE COMPUTER SYSTEMS, 2011,32(7):1310-1316.