A High-Speed Digital Signal Hierarchical Parallel Processing Architecture Based on CPU-GPU Platform

Yan Di, Shuai Weiyi
Equipment Academy
Beijing, China
e-mail: yandimail@126.com, 18803348431@163.com

Sun Ke
Beijing Aerospace Control Center
Beijing, China
e-mail: sk2220060710@126.com

Li Zibo
China Xi'an Satellite Control Center
Xi'an, China
e-mail: lizibo426@126.com

***Abstract*—Digital signal processing on the CPU-GPU platform has advantages of flexibility, scalability and easy maintenance. The powerful parallel processing capability of the CPU-GPU platform makes it inherently advantageous in high-speed signal processing. A hierarchical parallel processing architecture (HPPA) is proposed. This architecture uses a block processing technique different from stream processing used by field programmable gate array (FPGA). The main components are designed and the real-time performance of the architecture is analyzed. Conclusions can be used to determine the number of processing branches to ensure the real-time performance of the system.**

Keywords-CPU-GPU platform; HPPA; real-time; latency; throughput rate; block processing

I. INTRODUCTION

With the improvement of processing performance of CPU and GPU, it is possible to realize high-speed digital signal processing on CPU-GPU heterogeneous platform. In 2016 September, Intel released a new XeonPhi7200 family of multicore processors based on the many integrated core (MIC) architecture, called Knights Landing, with a maximum of 72 cores and 288 threads, clocked at 1.5GHz. In the same year, NVIDIA released the Tesla P100 card, which has 3584 processing cores, clocked at 1328MHz. The biggest advantage of the processor is that it can realize multi-threaded parallel processing, which is transplantable, low cost and scalable. It has been used in real-time signal processing. The related researches mainly focus on three aspects, such as software radar signal processing, software GPS receiver and software radio platform.

A. Software Radar Signal Processing

Reference [1] studied the method of using OpenMP to achieve strong real-time radar signal processing tasks under the X86 architecture CPU. Reference [2] and [3] studied the parallel strategy of task level, data level and thread level on CPU-GPU platform, and designed the parallel algorithm of radar signal.

B. Software GPS Receiver

In 2004, the United States Cornell University developed dual frequency GPS software receiver on the Pentium4 processor, using 80% of the computer resources to complete the processing of 10 channels [4]. In 2009, Thomas Hobiger of Japan implemented a GPU real-time software receiver in GPU environment (Intel Core2 Q9450 + NVIDIA GeForce GTX 280) [5]. In 2014, the German company IFEN released the receiver called SX3 [6], [7], which implemented 300 real-time tracking channels using CPU-GPU architecture in dell workstation.

C. Software Radio Platform

Related research mainly includes the open source GNU [8] Radio platform, and the SORA [9] platform developed by Microsoft Asia Research Institute. Both platforms use software pipeline to transform digital modulation problems into software programming problems. As the pipeline method depends on the CPU frequency, so GNU Radio processing rate can only reach tens of Kbps, SORA real-time communication rate is only 43.8Mbit/s. Obviously, high-speed digital signal processing can't be achieved based on these two platforms.

The purpose of this paper is to study a general-purpose parallel processing architecture on CPU-GPU platform and provide a template for the parallelization of digital signal processing. The main work includes two aspects, one is the proposed HPPA architecture, basic compositions of HPPA are designed, and the second is the real-time analysis of HPPA.

II. HPPA & COMPONENT DESIGN

A. Framework of HPPA

To meet the needs of high-speed digital signal processing, parallelization is carried out at two levels, data level and thread level, so as to form a hierarchical parallel processing

architecture as shown in Fig. 1. HPPA consists of five components and one parameter. The five components are signal segmentation, signal pool, process manager, processing branch and result combine. The parameter N_{branch} is the number of processing branches.

1) Signal segmentation

The function of signal segmentation is to decompose a one-dimensional serial data stream according to certain rules and form a plurality of signal blocks and fill them into a signal pool.

2) Signal pool

The signal pool is essentially a cyclic buffer, which ensures the stable operation of the processing system by eliminating the processing jitter of CPU and GPU.

3) Process Manager

The process manager is a scheduling center for the entire architecture, and its role is to distribute the signal blocks to each processing process. The idle process is discovered by monitoring the state of each process in real time, then the signal block is removed from the signal pool to be processed by the idle process.

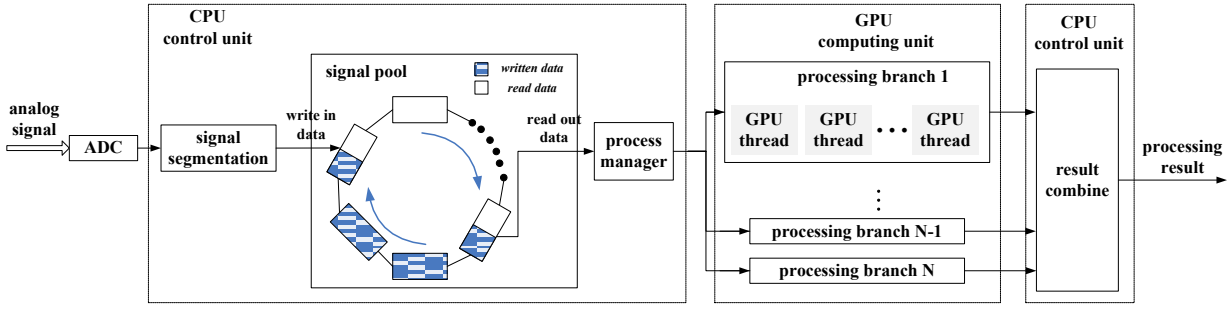

Figure 1. Framework of HPPA.

4) Processing branch

The processing branch is a process, and each processing branch is exactly the same except for the signal block to be processed.

5) Result combine

The function of result combine is to reprocess results of each processing branch according to certain rules and get the final results.

6) Number of processing branches N_{branch}

N_{branch} is an important parameter of HPPA, which determines the theoretical throughput rate of HPPA.

Signal segmentation, signal pool, process manager and result combine are all serial execution, involving a large number of logical operations, so it is suitable for running in CPU. The processing branch contains a large number of parallel operations, which is suitable for running in GPU. The architecture has the following two characteristics:

7) Hierarchical parallelization strategy

In order to give full play to the parallel processing ability of CPU-GPU platform, a hierarchical parallelization strategy of "data-level + thread-level" is adopted. Data-level parallelization refers to the creation of multiple processes in a signal processing task, each processing a signal block. The more the number of processes, the more blocks of signals that can be processed simultaneously. Thread-level parallelization refers to the use of parallel algorithms in each processing branch, the greater the number of threads used by each processing branch, the shorter the processing time of each signal block.

8) Block processing mode

The programmable hardware such as FPGA adopts flow processing technology. Under the control of global clock, the sampling points are processed in sequence. If this point by point method is still used in multi-core pipeline, the synchronization overhead and the loop overhead will be greatly increased. Therefore, block processing technology is used. As the name implies, block processing technology transforms the input dataflow into signal blocks. The block processing is shown in Fig. 2. T_{pro} is the processing time of a signal block, T_s is the sampling period, L is the number of sampling points of a signal block.

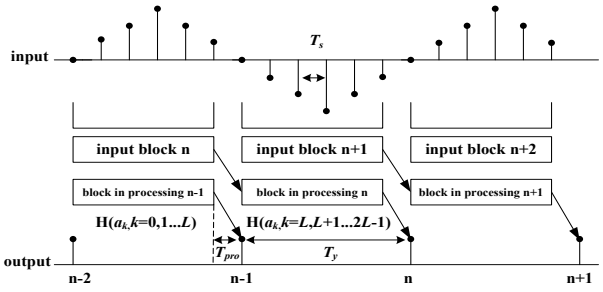

Figure 2. Sketch map of block processing.

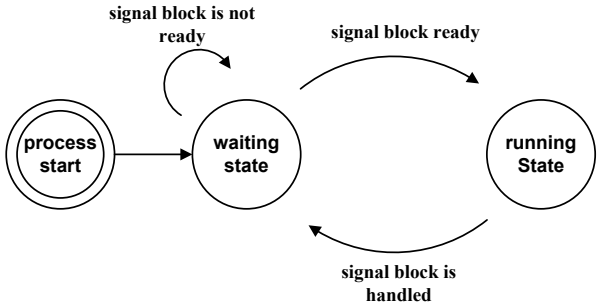

Figure 3. State transition diagram of a process.

B. Component Design

1) Processing branch

There are two states of waiting and running when the processing branch works. Waiting state means the process waits for the signal block to be processed. Running state means the process is handling the signal block. The status transitions for processing branches are shown in Fig. 3.

2) Process manager

The process manager is a CPU thread whose workflow is shown in Fig. 4. The process manager has been running in the loop since it started working and exited only when the task stopped command was detected.

3) Signal segmentation and result combine

Signal segmentation refers to dividing the signal stream into signal blocks of length L according to a certain method. The commonly used segmentation methods are shown in Fig. 5, critical segmentation and overlapped segmentation.

4) Critical segmentation

The critical segmentation means that there is no overlap between the adjacent two signal blocks.

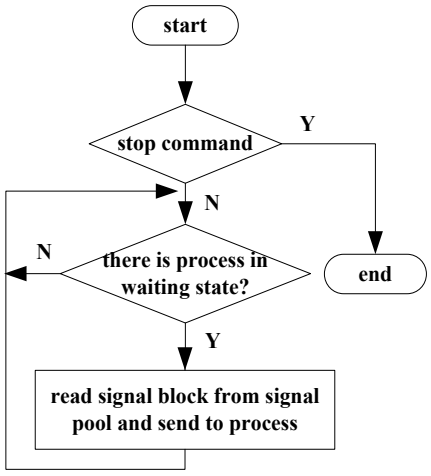

Figure 4. Workflow diagram of process manager.

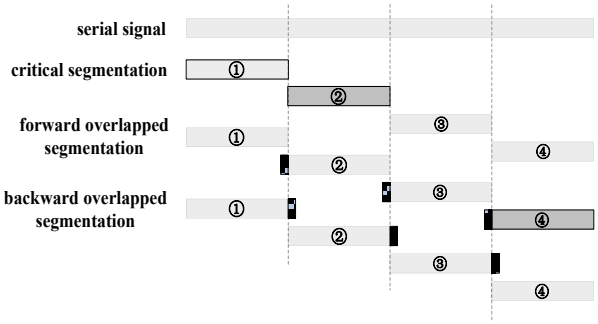

Figure 5. Signal segmentation methods.

5) Overlapped segmentation

Overlapping division means that the adjacent two signal blocks have overlapping parts. According to the position of the overlapping part, it can be divided into forward overlapping and backward overlapping. Forward overlap division means that the signal block overlaps with the previous signal block, and the backward overlap segmentation means that the signal block overlaps with the latter signal block.

III. Real-Time Analysis of HPPA

A. Real-Time Index

The most important problem of signal processing is real-time, throughput rate and delay are two important performance indexes.

1) Throughput Rate

The physical significance of the throughput rate is the number of samples that can be processed per second, expressed in Th, and the unit is samples/s. According to the different application environment, the throughput rate is divided into theoretical throughput rate and actual throughput rate.

a) Actual throughput rate Th_{actual}

Th_{actual} is the processing speed of an actual system, which can't be greater than the system sampling rate f_s. When $Th_{actual} < f_s$, the system is a non-real-time processing system.

b) Ttheoretical throughput rate Th_{theory}

Th_{theory} is the maximum processing speed of the system without restriction of sampling rate. When theoretical throughput is analyzed, the sampling rate is assumed to be infinite.

2) Latency

The latency refers to the time interval between the signal block and the processing result in a real running system, the unit is s.

B. Assumptions

In order to facilitate analysis, this paper makes three assumptions:

- In addition to processing branch, the effect of other components on the real-time performance of HPPA is not considered. The calculation of HPPA is mainly concentrated on processing branch. The main work of signal segmentation and signal pool is memory reading, and the main work of process manager and result combine is logical judgment, and the calculation is very small. Therefore, the real-time performance of processing branch basically represents the real-time performance of the entire HPPA.
- Different signal blocks have the same processing time. Due to the non-real-time operating system, reading and writing competition, bus competition and other factors, the processing time of different signal blocks is not equal, but the difference is very small. To facilitate analysis, we assume that each signal block has the same processing time.
- All threads of HPPA are not blocked by other events.

C. Analysis Results

The results of real-time analysis are shown in Table I. f_s is the sampling rate, T_s is the sampling period, $T_{pro}(L)$ is the processing time of a signal block, N is the serial number of the signal block, L is the number of sampling points of a signal block. We can get the following conclusions after analysis.

There is a threshold value η_N for the number of processing branches.

$$\eta_N = roundup(\frac{T_{pro}(L)}{LT_s}) \quad (1)$$

When $N_{branch} \geq \eta_N$, theoretical throughput rate is $\frac{N_{branch}L}{T_{pro}(L)}$, actual throughput rate is f_s, latency is $T_{pro}(L)$, the HPPA system is a real-time processing system. When $N_{branch} < \eta_N$, theoretical throughput rate is $\frac{N_{branch}L}{T_{pro}(L)}$, actual throughput rate is equal to theoretical throughput rate, latency increases with time, the HPPA system is a non-real-time processing system.

TABLE I. REAL-TIME ANALYSIS RESULTS

N_{branch}	conditions	Th_{theory} (samples/s)	>or=	Th_{actual} (samples/s)	latency (s)	real-time or not
1	$T_{pro}(L) < LT_s$	$\frac{L}{T_{pro}(L)}$	>	f_s	$T_{pro}(L)$	Yes
	$T_{pro}(L) > LT_s$	$\frac{L}{T_{pro}(L)}$	=	$\frac{L}{T_{pro}(L)}$	$NT_{pro}(L)-(N-1)LT_s$	No
2	$T_{pro}(L) < LT_s$	$\frac{2L}{T_{pro}(L)}$	>	f_s	$T_{pro}(L)$	Yes
	$LT_s < T_{pro}(L) < 2LT_s$	$\frac{2L}{T_{pro}(L)}$	>	f_s	$T_{pro}(L)$	Yes
	$2LT_s < T_{pro}(L) < 3LT_s$	$\frac{2L}{T_{pro}(L)}$	=	$\frac{2L}{T_{pro}(L)}$	$\frac{N+1}{2}T_{pro}(L)-(N-1)LT_s$, *N is odd* $\frac{N}{2}T_{pro}(L)-(N-2)LT_s$, *N is even*	No

IV. CONCLUSION

It is feasible to perform high-speed digital signal processing on the CPU-GPU platform. In this paper, a hierarchical parallel processing architecture is proposed, which is composed of five parts: signal pool, processing branch, process manager, signal segmentation and result combine. The real-time performance of HPPA is analyzed from two aspects of throughput rate and latency, and the conclusions can be used to determine the number of processing branches. The next step is to implement the architecture with specific signal processing algorithms.

REFERENCES

[1] Wei Mengyao. Research on radar signal processing based on the X86 CPU[J]. Electronic Science and Technology, 2017, 30(5):55-57.

[2] Qin Hua, Zhou Mo, Cha Hao, et al. Research on multi-GPU parallel technology in software radar signal processing[J]. Journal of Xidian University, 2013, 40(3):145-151.

[3] Qin Hua, Zhou Mo, Cha Hao, et al. Research on the parallel technology of GPU acceleration on radar signal processing[J]. Ship Science and Technology, 2013, 35(7):77-82.

[4] Wu Xinbo. Research on GPS Software Receiver Based on CUDA[D]. Beijing Institute Of Technology, 2015.

[5] Hobiger T, Gotoh T, Amagai J, et al. A GPU based real-time GPS software receiver[J]. Gps Solutions, 2010, 14(2):207-216.

[6] Pany T, Riedl B, Winkel J. Efficient GNSS signal acquisition with massive parallel algorithms using GPUs[C]//Proceedings of the 23rd International Technical Meeting of the Satellite Division of the Institute of Navigation. 2010: 1889-1895.

[7] Falk N, Hartmann T, Kern H, et al. SX-NSR 2.0–A Multi-frequency and Multi-sensor Software Receiver with a Quad-band RF Front End[C]//Proceedings of the 23rd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2010). 2001: 1395-1401.

[8] BLOSSOM E. GNU radio: tools for exploring the radio frequency spectrum[J]. Linux journal, 2004, 2004(122): 76-81.

[9] TAN K, LIU H, ZHANG J, et al. Sora: high-performance software radio using general-purpose multi-core processors[J]. Communications of the ACM, 2011, 54(1): 99-107.