FPGA Implementation of High Performance and Low Power VOD Server

Qiang Cao, Jiang Jiang, Chang Wang, and Yongxin Zhu

Abstract—**Video play has become a popular entertainment as network capacities have been improved. To lower the power of the video stream server to meet the requirements of green computing and to save the bandwidth which helps to solve the bandwidth congestion problem, a novel hardware-based VOD server system is introduced. We design an architecture that is suitable for video stream transmission. We manage to complete a prototype system which is implemented in an FPGA board. In this paper, we describe the structure of the VOD system, the method of processing requests and the organization of large amounts of information. Besides, we propose a dynamic file chunking technique to implement real time playing. With it, users can interact with the system when playing a video.**

***Index Terms*—VOD, FPGA, architecture, low power.**

I. Introduction

The development of broadband internet technologies promotes the emergence of numerous web applications. But web applications are no longer limited to the transmission of data text and graphics. Multimedia videos and audios have become an important role of the web application. The VOD [1]-[3] (video on demand) system which is used for entertainment, education and advertisement has become a hotspot application. VOD technology which is based on real time media streaming [4]-[6] technology makes it possible to watch a video at any time. VOD can be either HTTP-based [7] or RTSP/RTP-based [8]. It's a good choice to use the HTTP protocol because it's popular and simple.

In recent years, many researches have been done on VOD while few researches have been done on hardware-based VOD. However, hardware-based TCP/IP protocol stack [9] [10] has been studied and been implemented by many researchers. This reveals that it's a trend implementing hardware-based servers. Considering the development cycle and development cost, the FPGA is a good choice. A web server on FPGA is implemented in [11].

HTTP protocol has been widely used for many network applications, including VOD. Users can send HTTP request to download videos from the media server and watch them online. Most of the current HTTP live video solutions are based on HTTP requests/responses. A video file is divided into a set of small fragments. A client sends an HTTP request for a specific fragment and receives the fragment via an HTTP response from the media streaming server. After playback of the fragment is nearly completed, the client sends a request for the next fragment until all the video data is send [12]. If the client skips some specific fragments, those fragments will not be transmitted. It helps to save the bandwidth. Every fragment is transmitted as a single streaming and is send from the start of the fragment. So the delay is unavoidable when a live event happens. To improve this condition, dynamic chunking technique is used to produce dynamic fragments.

With the development of broadband internet technologies, the 10Gbps physical Ethernet cards become main streaming. Along with the clouding computing, how to lower power consumption becomes an important issue. Based on the hardware design, a VOD server can achieve good performance as well as low power.

II. Basic Architecture

The simplified architecture of the hardware-based VOD server system is shown in Fig. 1.

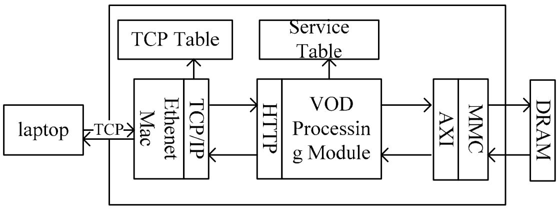


Fig. 1. Simplified architecture of the hardware-based VOD system.

HTTP requests from clients in the form of TCP packets are sent over the network. The 10Gpbs Ethernet media access control (MAC) receives the TCP packets. Then the TCP/IP module filters the requests and passes the legal ones to VOD processing module. The VOD processing module (VPM) adds a new entry to the service table. The VOD processing module deals with every entry in a circular manner. It retrieves the data from the DRAM with advanced extensible Interface (AXI) bus and passes the data to the TCP/IP module. The overall architecture of the VOD server system is shown in Fig. 2.

The system mainly contains four parts: a MicroBlaze processor core, a TCP/IP module, storage hierarchy and VPM. The MicroBlaze processor runs no operation system. It's used to initialize the whole system and control the DMA.

The TCP/IP module is written in Verilog hardware description language (Verilog HDL). It's a simplified implementation of the TCP/IP protocol. It implements the data link layer, the network layer and the transport layer. The functions of the TCP/IP module include ARP (Address Resolution Protocol) response, accepting TCP (Transmission Control Protocol) connections, keeping TCP connections and cutting off TCP connections.

Manuscript received August 20, 2013; revised October 31, 2013.

The authors are with the School of Microelectronics, Shanghai Jiao Tong University, Shanghai, China (e-mail: qiangcaohao@sina.com).

DOI: 10.7763/IJFCC.2014.V3.286

The storage hierarchy is made up of BRAM, DRAM, and SSD. The control information is stored in BRAM. DRAM is used as a data cache. SSD is used as large capacity storage. VPM realizes a video-on-demand application. It's the core of the study. The detail architecture of the VPM is shown in Fig. 3.

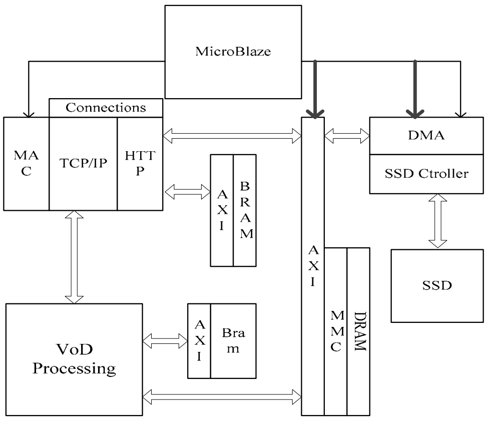


Fig. 2. Overall architecture of the VOD system.

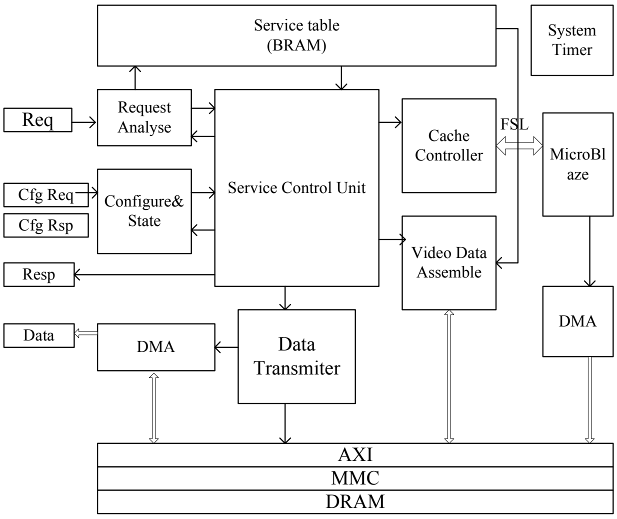


Fig. 3. Detail architecture of VPM.

The Service Control Unit which is the center of the VPM controls all the other modules to function parallel so that the design improves the performance of VOD application. At the other head, instead of using an embedded microprocessor whose efficiency is low to control the all modules, we only use it to control the DMA. It is hard to control the DMA by hardware and the DMA is not so important in this system. As a result, it shortens the response time and improves the throughout.

The whole system is implemented by FPGA whose power consumption is so low that the hardware based VOD server consumes low power.

III. Implementation

A. VPM and Storage Hierarchy

VOD related requests and configuration requests are the two kinds of requests that VPM will deal with. The Request Analysis Module receives HTTP requests and transforms the requests into easy to use information. And then it tells the Service Control Unit (SCU) to store the information synchronously. An item in Service Table is shown in Table I.

TABLE I: Item Organization in Service Table

Parameter	type	description
connection id	uint32	identify of the TCP connection, generated by TCP/IP module
request file name	char[12]	the multimedia file name on command, assume maximum length of file name is 12
param1	uint16	the start point of the file useful only when partial file wanted, otherwise, fill 0
param2	uint16	the end point of the file useful only when partial file wanted, 0 represented the end of the file
state	uint16	service state
data length	uint32	total length of data should be transmit in this session
ast access time	uint32	timeout if the service has no response for a long time
request time	uint32	time the request comes, for rate control
data offset	uint32	data transmission pointer
next	uint32	when using link table, this parameter represents the next service item
DATA1	uint64	data address and length,33 high-bit for address and 31 low-bit for length, 0 for no data
DATA2	uint64	the same as above
DATA3	uint64	the same as above
bytes send	uint32	data bytes already send from connection to now
max data rate	uint16	bandwidth control

A finite state machine which interacts with other part of VPM exists in SCU for coordinate work. The flow how the SCU deals with a request is shown in Fig. 4.

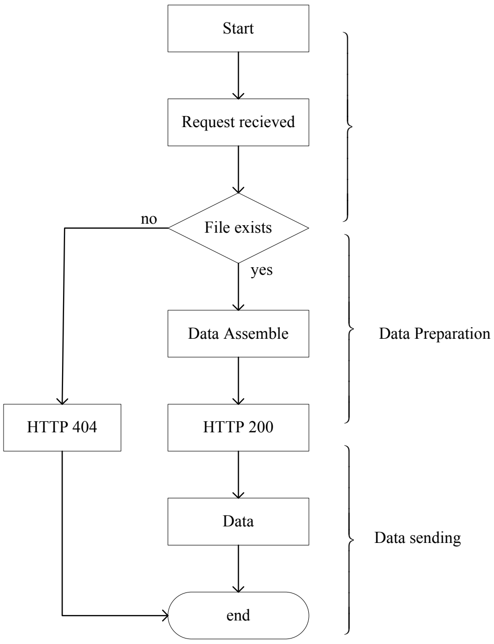


Fig. 4. Flow how to deal with a quest.

It can be divided into three stages: request receiving, data preparation and data sending. The Request Analysis module completes the first stage. The second stage is completed by the Cache Controller and the Video Data Assemble module. The last stage is completed by the Data Transmitter.

There is so much data that how to organize the storage is an important issue. The Xilinx FPGA VC6VLX550T is chose and the storage elements are show in Table II. The memory is assigned in accordance with Table III.

Unlike the VOD request, the configuration requests don't use the HTTP protocol so that the requests can be deal with quickly. Most of all, we only read registers by sending configuration requests although we can both read and write registers.

TABLE II: AVAILABLE STORAGE ELEMENTS

Type	size(Kbit)
Distributed RAM	6,200
Block RAM	22,752
DDR3 SDRAM(EX)	67,108,864

TABLE III: STORAGE ELEMENTS USED

unit	Type	size(byte)
MicroBlaze	DRAM	1G(exclusive),4G(addressable)
TCP/IP	BRAM	1M
VPM	BRAM	128K
	DRAM	1G
Video Data Cache	DRAM	6G
FIFO	BRAM	64K~128K

B. Cache Policy

The data cache whose physical media is dynamic random access memory and whose size is 6G bytes is used to speed up data access. Hash algorithm is used to map the file to the physical cache blocks. Least recently used method (LRU) is used to replace the cache block when hash conflict happens because hot data is preferred to cache. Asymmetry 2-way associative cache is used. The hash conflict rate is shown in Fig. 5. The horizontal axis represents the set 2 ratio.

$$Set2ratio = \frac{setblock2}{setblock1 + setblock2}$$

The vertical axis represents the hash conflict rate. The cache block sizes vary from 256k byte to 4M bytes. The least hash conflict distribution is shown in Fig. 6.

C. Real-Time Transmission and Data Bandwidth Control

To implement real time video playing file chunking technique is used. A video file can be dynamically split into multiple pieces, according to the key-frame stamps. Unlike static file chunking technique which splits the video file in advance, dynamic file chunking technique can help to split a video according to a request. Thus, the split file piece can start and end at any point of a file. Two parameters, param1 and param2 shown in Table II, are delivered when a request is send. If a video file has key frame stamps, it's easy to find the wanted start and end point. Only the specified file part will be transmit.

There are three parameters in Table II related with data bandwidth control. There is another global default parameter called global maximum data rate related with bandwidth control. If the max data rate is invalid, the global maximum data rate is used. This parameter can be configured when the system is initialized. It can be configured through configuration module as well. A video may be stopped at any time as it is playing. So it's not a good idea to download the whole video as quick as possible. We can change the download rate according to the request. The parameter max data rate is used to control this. And it should meet the following condition.

$$datarate = \frac{bytessend}{currenttime - requesttime}$$

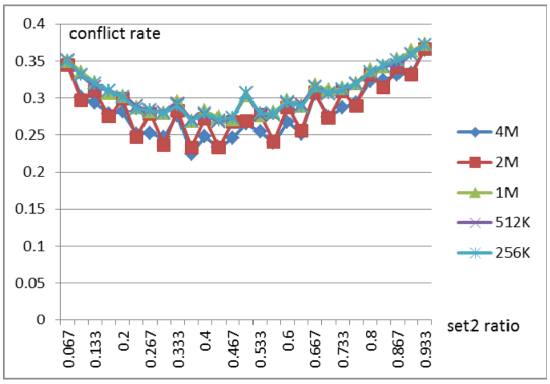


Fig. 5. Hash conflict rates in different block size and set-2 ratio.

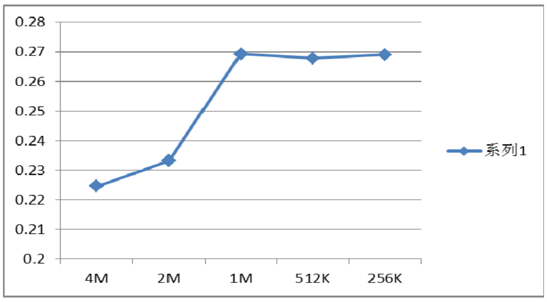


Fig. 6. Least hash conflict rates in different block size.

D. User Interaction

When playing a video, a user can send the following requests in the Table IV. Some of the requests are realized in a trick way because they are difficulty or unnecessary to realize. However, it seems to realize all the commands in the eyes of users.

TABLE IV: USER INTERACTION REQUESTS

command	description	realization
play/Resume	Start a video from the beginning or resume after temporarily stopping the show	start transmit data
stop	permanently stop the presentation of the video	stop data stream and cut off the connection
pause	Freeze the picture	pause transmit data temporarily
jump forward/backward	Jump to a particular time in the presentation	stop the former stream and transmit the video from the particular time
fast forward/slow down	going forward at a higher/lower rate than normal	change data stream rate and playback the video data in fast/slow mode

IV. PERFORMANCE EVALUATION

The platform of our design is a custom board with four FPGAs shown in Fig. 7. Only one of the four FPGAs is used.

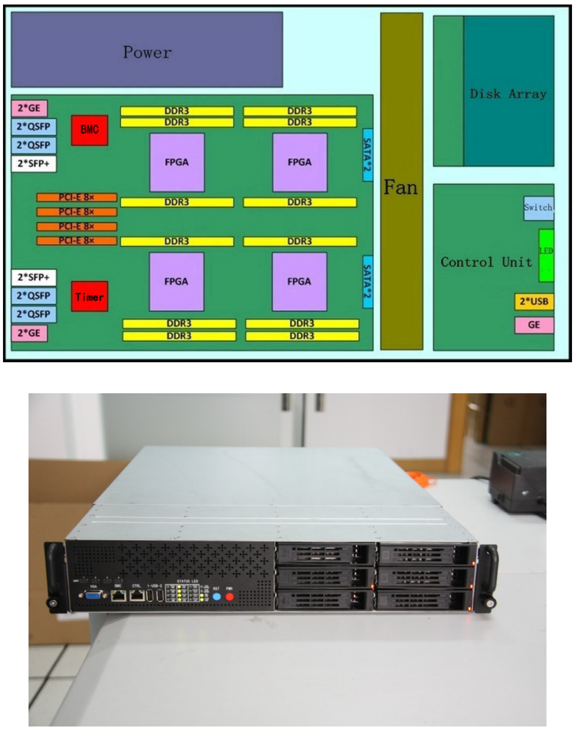


Fig. 7. custom board with 4 FPGAs.

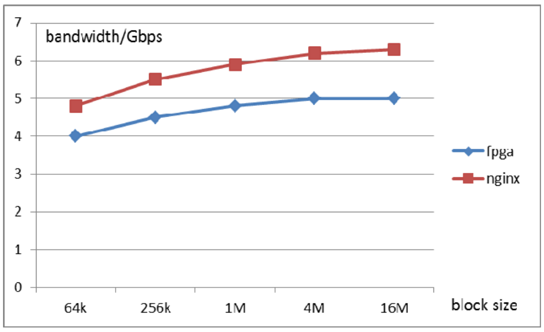


Fig. 8. Throughputs.

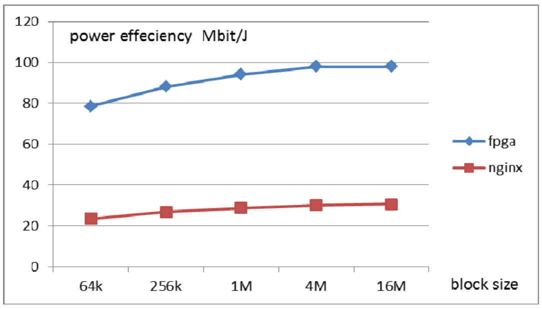


Fig. 9. Power efficiency.

On this board, The Xilinx FPGA VC6VLX550T is used. We use Xilinx ISE Design Suite 14.1 which this newest a new version when we start the project. To construct the system, we use some the provided IPs by Xilinx, including MIG Virtex-6 and Spartan-6 3.91, XAUI 10.3 and Ten Gigabit Ethernet MAC 10.3.

The system implements on FPGA runs at a rate of 151MHz while the Microblaze runs at 100MHz and the MAC runs at 156.25MHz. The size of the DDR3 SDRAM on the FPGA board is 8Gbytes.

The system is compared with Web test equipment, Spirent Test Center. Results are compared with Nginx which runs on a 6-core 12 thread processor, Inter Xeon series. The size of its DDR3 memory is 16Gbytes. The speed of the physical Ethernet port on the FPGA board is 10 Gbps as well as on the Inter Xeon platform. All the video data are present in the DDR memory of the testing system as well as the main memory of the reference IBM platform. The results are shown in Fig. 8.

Although the throughputs are not good enough to be compared with the Xeon platform for the reason that it’s just a prototyping system and the clock frequency is low, its power is much lower than the reference platform. The power of the FPGA system is about 51W while the reference platform is about 205W. The power efficiency is shown in Fig. 9.

V. Conclusion And Future Work

In the design of this paper, we show a hardware-based VOD system implemented on FPGA. We show how to response the request of users, how to realize the real time transmission and how to control the video rate to save the bandwidth. We use dynamic file chunking technique instead of static file blocks to avoid the unnecessary data transmission. As a result, we shorten the response delay. We use FPGA to complete most of the task and achieve a better power efficiency although a MicroBlaze processor is used. The throughputs are a bit lower than Nginx which runs on Xeon platform because the system runs at low clock rate and the bus delay is unavoidable. However, the power efficiency is higher than Nginx running on Xeon platform. Still, there is space to improve the performance. It is possible to prompt the performance near the limit of 10Gbps if we carefully optimize the memory access. With more work to be done, we believe the hardware-favored system can have better performance and save more energy.

Acknowledgment

This paper is partially sponsored by the National High Technology Research and Development Program of China (863 Program) (No.2009AA012201) and National Natural Science Foundation of China (No.61373032).

References

[1] J. Y. B. Lee, “On a unified architecture for video-on-demand services,” *IEEE Trans. Multimedia*, vol. 4, no. 1, pp. 38-47, 2002.

[2] S. A. Yang, H. Yang, and Y. H. Yang, “Architecture of high capacity VOD server and its prototype implementation,” *Shanghai Jiaotong Daxue Xuebao*, pp. 2082-2085, 2004.

[3] J. W. Kim, “Effective video block placement scheme on VOD server based on multi-zone recording disks,” in *Proc. International Conference on Multimedia Computing and Systems*, pp. 29-36, 1997.

[4] Z. G. Qin, “Flourishing in the internet: The huge success of live peer-to-peer media streaming,” in *Proc. ICCASM 2010 - 2010 International Conference on Computer Application and System Modeling*, 2010, vol. 11, pp. V1176-V1180.

[5] X. H. Hu, “Evaluating and enhancing the performance of streaming media services in E-learning system,” *Advances in Intelligent and Soft Computing*, vol. 163 AISC, pp. 205-211, 2012.

[6] G. Cheung, “On media data structures for interactive streaming in immersive applications,” *The International Society for Optical Engineering*, vol. 7744, 2010.

[7] B. Niels, “An autonomic delivery framework for HTTP adaptive streaming in multicast-enabled multimedia access networks,” in *Proc. IEEE International Conference on Multimedia and Expo*, pp. 1248-1253, 2011.

[8] J. W. Bang, “Efficient RTSP-based multiple buffering and packet transmission methods for delivering OMA PoC Box service,” *Computer Networks*, vol. 56, no. 15, pp. 3468-3478, October 15, 2012.

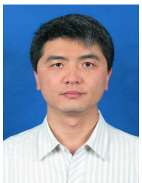
[9] R. B. Wang, “Reduced TCP/IP protocol implement in VHDL,” in *Proc. 2009 International Workshop on Intelligent Systems and Applications*, pp. 1-5, 2009.

[10] Y. Ji, “40Gbps multi-connection TCP/IP offload engine,” in *Proc. 2011 International Conference on Wireless Communications and Signal Processing*, pp.1-5, 2011.

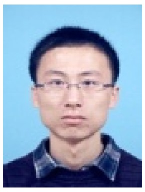
[11] J. B. Yu, “Grounding high efficiency cloud computing architecture: HW-SW co-design and implementation of a stand-alone web server on FPGA,” in *Proc. 4th International Conference on the Applications of Digital Information and Web Technologies*, pp. 124-129, 2011.

[12] G. Giovanni, “Low-Latency live video streaming over low-capacity networks,” in *Proc. 8th IEEE International Symposium on Multimedia*, pp. 449-456, 2006.

Qiang Cao received his B.S. in Microelectronics from Shanghai Jiao Tong University, China in 2011. From 2011 to now, he works as a graduate in School of Microelectronics, Shanghai Jiao Tong University for his M.S. in computer science and technology. His mainstream research interests include FPGA and embedded systems, algorithm accelerator, and digital circuit design.



Jiang Jiang received his Ph.D. and M.S. degrees in computer science and technology from National University of Defense Technology, Changsha, China in 2002 and 1998, respectively. He received his B.S. in computer from National University of Defense Technology, Changsha, China in 1995. From 2009 to 2010, he works in University of Toronto as a visiting scholar. From 2002 to 2010, he works as a researcher in Institute of Computer Science, National University of Defense Technology, Changsha, China. His mainstream research interests include multi-core/may-core microprocessor architecture, high-performance accelerator, and Artificial Neural Networks. In 2006, he joined the Galaxy-x custom CPU project as a team leader and made a significant contribution. From 2010, he works as an associate professor in School of Microelectronics, Shanghai Jiao Tong University.



Chang Wang received his M.S. degree from the School of Computer Science, Wuhan University, Wuhan, China, in 2010 and his B.S. degree from the school of computer science and technology, Xi'an Polytechnic University, Xi'an, China in 2006. He is currently furthering his study in the Microelectronic School of Shanghai Jiao Tong University, Shanghai, China as a doctoral student. His research interests include microprocessor architecture and artificial neural network.