An Implementation of Rate-Adaptive IPTV System on DSP

Yen-Jen Chen, Chih-Chung Wang, and Jr-Jen Huang

***Abstract*—This paper proposes an implementation of rate-adaptive IPTV system on DSP. The system has a client-server based architecture. The server-side and client-side are communicated with network socket connection. The server-side hosting on the DSP platform encodes the video signal from the TV cable. The client-side using JavaScript Language decodes and plays the multimedia video of DSP server-side instantly. The video quality mechanism is controlled by network bandwidth detection to improve the video play quality. The client-side detects the used bandwidth of the video stream from the server-side. The bandwidth utilization information is sent to the server-side to adjust the video encoding rate. The DSP platform used for this work is TMS320DM6437 EVM, produced by TI, implementing H.264 and AAC Encoders. The Encoder program uses the codec source code wrapped by TI, which conforms to the xDAIS (eXpress DSP Algorithm Interoperability Standard), and it is integrated with TI DSP network developer's kit and TV tuner to implement a TV over the Internet with the channel selection control.**

***Index Terms*—IPTV, H.264, AAC, DSP, Javascript, Tuner**.

I. Introduction

As the network and multimedia technologies develop rapidly, the present digital service on Internet has become diversified, so that the users can enjoy various multimedia and information services. The digital application service is defined as "technology and service of digitizing video and audio data, and integrating them", the Internet Protocol Television (IPTV) is one of popular applications providing digital application service.

The digital signal processor is used in an extensive area, including advanced multimedia applications, such as cell phone, MP4, mobile video, etc. The above applications and recently popular video compression encoding technique require massive and complex arithmetic processing. Therefore, in order to provide real-time, smooth and convenient services, this study uses a new generation video compression technique H.264 and advanced audio coding, implemented in DSP, and uses the client-side of Javascript program language and DSP server-side for network socket connection, as well as TV channel selection control.

The real-time multimedia video sent from DSP server-side is received, decoded and played. The video quality mechanism is controlled by network bandwidth detection to improve the video play quality.

Manuscript received September 20, 2012; revised December 17, 2012. This work was supported in part by R.O.C. (Taiwan) National Science Council under Grant NSC 100-2221-E-131-003.

Yen-Jen Chen, Chih-Chung Wang, and Jr-Jen Huang are all with the Department of Electronic Engineering, Ming Chi University of Technology, Taiwan, R.O.C. (Email: yjchen@mail.mcut.edu.tw, a6116122@mail.mcut.edu.tw, jrjen@mail.mcut.edu.tw).

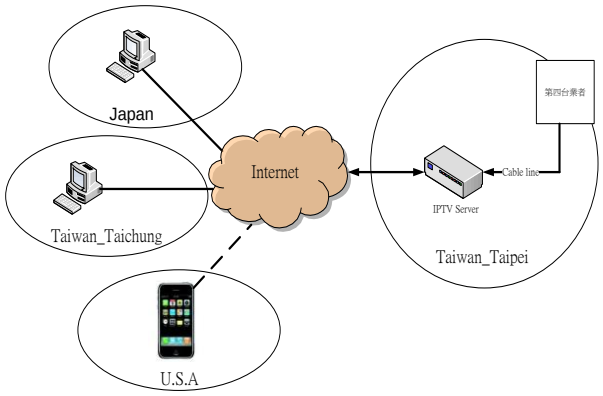

Fig. 1. Schematic diagram of IPTV service

II. Background Knowledge

A. H.264 Advanced Video Coding

H.264 is a standard for video compression, and is currently one of the most commonly used formats for the recording, compression, and distribution of high definition video. The final drafting work on the first version of the standard was completed in May 2003.

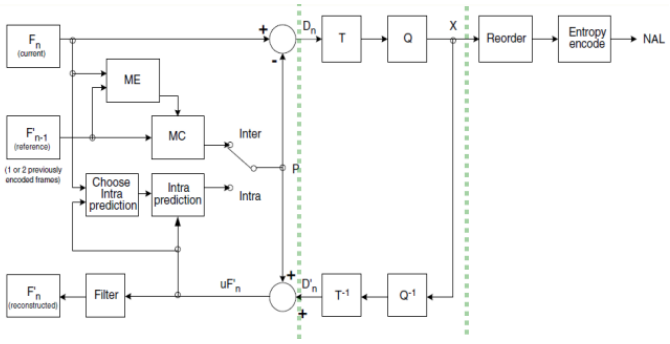

Fig. 2. The H.264 video codec process [1]

An H.264 video encoder carries out prediction, transform and encoding processes (see Fig. 2.) to produce a compressed H.264 bitstream. An H.264 video decoder carries out the complementary processes of decoding, inverse transform and reconstruction to produce a decoded video sequence

B. H.264 Video Encode

In terms of the picture quality level of H.264 bitrate, as shown in TABLE I, higher bitrate indicates better picture quality. When the H.264 bitrate is less than 500kbps, the picture quality is worse. When the bitrate is 500kbps to 1500kbps, the picture quality is equivalent to that of MPEG-1 level. When the bitrate is 1500kbps to 3000kbps, the picture

DOI: 10.7763/JACN.2013.V1.3

quality is equivalent to that of MPEG-2 level. When the bitrate is 3000kbps to more than 5000kbps, the picture quality is equivalent to that of HDTV (High Definition Television) 720p and 1080i levels.

TABLE I: H.264 BITRATE PICTURE QUALITY LEVELS

H.264 Bitrate	H.264 QP	Directions
Less than 500kbps	Less than 25	The worst quality
500~1500kbps	25~34	The MPEG1 quality
1500~3000kbps	35~40	The MPEG2 quality
3000~5000 kbps	41~45	The HDTV-720 quality
More than 5000 kbps	More than 45	The HDTV-1080 quality

C. AAC Audio Encode

Designed to be the successor of the MP3 format, AAC generally achieves better sound quality than MP3 at similar bit rates.AAC was developed with the cooperation and contributions of companies including AT&T Bell Laboratories, Fraunhofer IIS, Dolby Laboratories, Sony Corporation and Nokia.

AAC supports inclusion of 48 full-bandwidth (up to 96 kHz) audio channels in one stream plus 16 low frequency effects channels, The quality for stereo is satisfactory to modest requirements at 96 kbit/s in joint stereo mode; however, hi-fi transparency demands data rates of at least 128 kbit/s (VBR). AAC is also the default or standard audio format for YouTube, Sony Ericsson, Nokia, Android, Apple mobile phones with the use of a converter.

Three common specifications of AAC are Main Profile, Low Complexity Profile and Scaleable Sampling Rate, the technologies are compared in Table II.

TABLE II: AAC SPECIFICATIONS COMPARISON

Profile	Module option
Main Profile	Not use Gain control Module Levels of TNS is 20 (highest) Use Prediction Module
Low Complexity Profile(LC)	Not use Gain control Module Reduce T N S levels to 12 Cancel Prediction Module
Scaleable Sampling Rate Profile (SSR)	Use Gain control Reduce T N S levels to 12 Cancel Prediction Module

The sound quality of main profile is slightly different from that of low complexity profile, but the coding time and efficiency are different obviously. Therefore, the low complexity profile is widely used at present. In order to reduce the complexity and DSP load, this experiment uses the digital audio source for IPTV in low complexity profile format [2].

D. TMS320DM6437 EVM Development Environment

This experiment environment uses TMS320DM6437 EVM (Evaluation Module) of TI as the development platform. It is of DaVinci TM processor family [3]. The EVM means a complete hardware development module is provided for developers. The video module and memory module for this experiment have available peripheral functional equation libraries. In addition, the hardware design adopts VelociTI architecture, each clock can execute at most eight 32-bit instruction sets, and has a pipeline [4]. They are the main characteristics of DSP.

E. TMS320DM6437 EVM Hardware Platform

The hardware architecture inside *TMS320D*M6437 EVM development platform is shown in Fig. 3. Symbols A-I represent the required hardware blocks for this experiment, and the hardware components are briefly described according to the symbols.

1) Texas Instruments DM6437 Processor, speed 600MHz.
2) TVP5146M2 video decoder, including CVBS signal A/D Transform, input end supports Composite-video and S-video specifications.
3) Containing two 64MBbytes DDR2, 128 Mbytes in all.
4) 16 Mbytes of non-volatile non-volatile flash memory, the compiled program is written in the access position.
5) Embedded JTAG emluator interface, transmission can be executed without high priced JTAG, USB can be used as transmission interface.
6) Power input end, 5V and 3A transformer is supported.
7) The Ethernet 100M/bps RJ-45 Port.
8) AIC33 audio codec, input end supports AUX specification.
9) PCI slot interface, this interface can be connected to the I/O of various chips on the development board for extra development and applications.

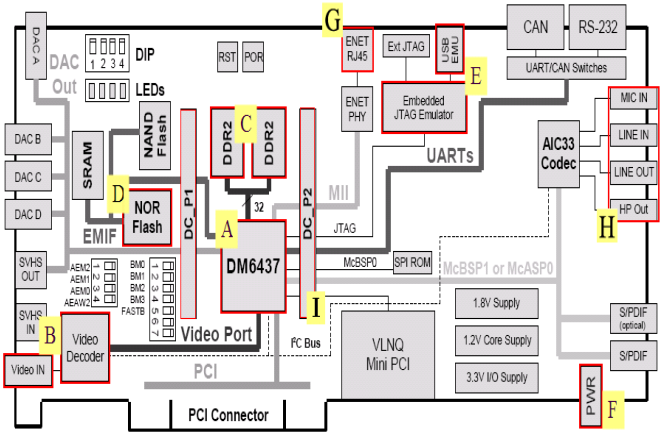

Fig. 3. Block diagram of hardware inside DM6437 EVM [5]

F. TV Tuner Module

The TV tuner module is used in this experiment in order to enhance the integrity, so that the users can select channels instantly on the network. The TV tuner module is a circuit composed of TNF5535MF [6] core of TNF company, using I ^{2}C pins SCL, SDA and DM6437 for connection. When the users select a channel, DM6437 transmits the channel parameter via I^{2}C to TNF5535MF. When the TNF5535MF receives the channel parameter, it executes channel selection immediately, and divides the correct channel video CVBS and audio AF signals into two. The signals are exported to the

AV and AUX terminals of circuit, and the terminals are connected via transmission line to the Video single in and audio single in of TMS320DM6437 EVM, so that the DM6437 executes digital video coding, as shown in Fig. 4.

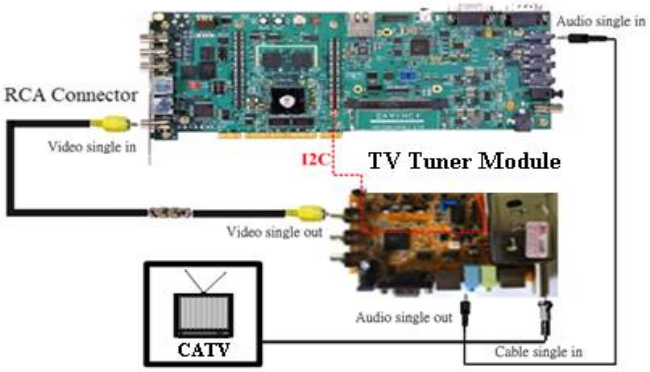

Fig. 4. TMS320DM6437 EVM and tuner connection architecture

III. Implementation Method

A. Server-Side Experimental Architecture

The server-side system architecture is shown in Fig. 5. The video processing action is cut to five major blocks, video input, video A/D, video encode, video decode and video output, in charge of video in, encode, decode and video out tasks, respectively. The audio processing action is cut to four major blocks, audio input, speech encode, speech decode and audio output, in charge of audio in, encode, decode and audio out tasks, respectively. The file processing is divided into four major blocks, video and audio file output and file input, in charge of file output and input. The middle control block is in charge of the control and communication among all the blocks. Eventually, the above blocks are connected to form the overall system.

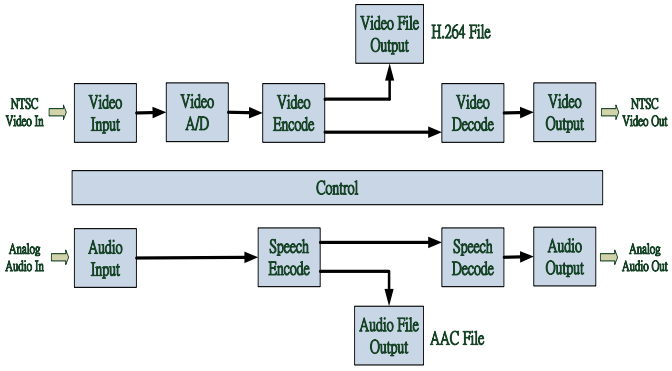

Fig. 5. Server-Side system architecture

In terms of video encode, the NTSC video enters the video A/D block and is converted into YUV format, enters the video encode block, then the H.264 encode calculation wrapped by TI conforming to the xDAIS (eXpress DSP Algorithm Interoperability Standard) [7] is executed in DSP. It is one of procedures producing the highest speed in computation of DSP. When the H.264 encode processing is completed, the file output block establishes network connection, and cuts data to various network packets. The data are transmitted to the client-side to complete the procedure. As for another procedure, the video output block is entered from video decode, exported to the video output terminal composite-video or S-video on the development board.

The next step is the audio processing procedure. When the audio is imported into the audio input block for access data, transmitted to the speech encode block for AAC Encode action. In this system, the AAC Encode mode is used, the codec is wrapped by TI following xDAIS. It can be transmitted to the file output block after Encode is finished to prepare for network transmission. When the client-side and server-side are connected, the client-side receives network packets. Another procedure is to enter the speech decode to speech output block, exported to the tone source output terminal AUX on the development board. Thus, it is the complete simultaneous work flow of video and audio.

B. Client-Side Experimental Architecture

The DM6437 development board platform is introduced above. Besides good hardware platform and experimental environment, designing a simple and convenient graphic user interface is more important, and it must be able to communicate with the hardware development platform. The presentation of the final user interface control side should enable the operator to control the server-side remotely quickly only by clicking and entering basic parameters, as shown in Fig. 6.

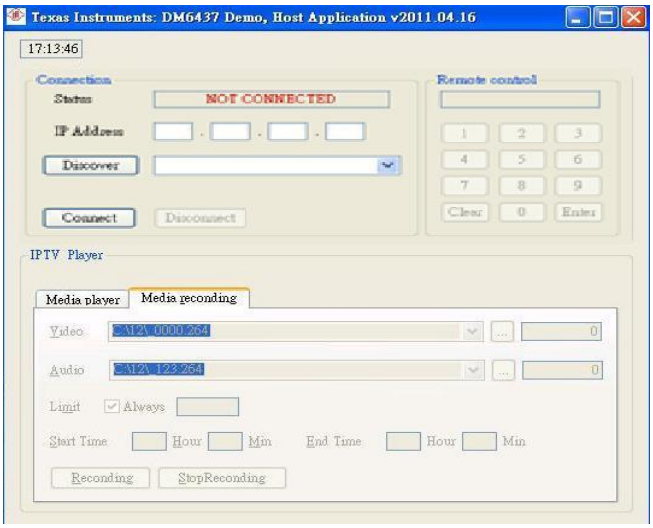

Fig. 6. User interface control side

The GUI used by the client-side is the Standard Widget Toolkit technique created by IBM, which supports Java operating system. The graphic interface is created by accessing the Native Graphical User Interface Libraries in the operating system and using the Java Native Interface. The graphic interface is created before the main Rhino JavaScript Engine of program development. This plan is maintained by Mozilla, aiming to implement Server-Side JavaScript, so that the JavaScript can be implemented in Java.

In this experiment, JavaScript is extensively used in script language of Web client-side, it is mainly used in dynamic webpage technique to create more diversified operation interaction, and it does not provide massive library for the users to access file systems, or to implement network communication protocol and create graphic interface. Thus, the Rhino JavaScript Engine must be used, so as to implement JavaScript in Java. The network library, file access library and

GUI library of Java are used in this program development process. The source code program files are divided into Main.js, Rpc.js, Fileio.js and Ipc.js. The program architecture relation is shown in Fig. 7.

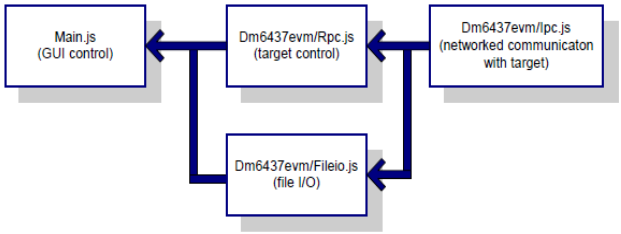

Fig. 7. User control side program architecture [8]

The **Main.js** is in charge of the GUI only. It draws buttons and input fields and listens for user actions to send an appropriate command to the target (for example, start playing, stop playing, and switch mode).

The **Rpc.js** (Remote Procedure Call) provides the GUI script with several functions, of which the important ones are:

1) rpc.connect : Connects to the target at ip Address.
2) rpc.paramGet: Gets the state of a target parameter
3) rpc.paramSet: Changes the state of a target parameter.
4) rpc.controlPlay/Stop: Starts and stops the target's audio /video processing.

The **Fileio.js** script has an even simpler API:

1) fileio.connect: Connects to the target's fileio port.
2) fileio.recvCmd: Receives File I/O commands from the target.
3) fileio.dispatchCmd: Runs received File server commands

The **Ipc.js** (Inter-Process Communication) takes data structures from Rpc and Fileio, serializes them, and sends or receives them to or from the target over the network.

Therefore, the basic process of program is that the Main.js is the entry point of program initialization, the network communication and file input and output capabilities are obtained by uploading Rpc.js and Fileio.js. Therefore, the Main.js uses Standard Widget Toolkit to draw various graphic components to create the graphic interface, and the Java program language and JavaScript script language write the program codes responding to the actions of various buttons, options and windows by reading Rpc.js and Fileio.js. Finally, Rpc.js and Fileio.js read Ipc.js for network communication and file access with the DM6437Server service side. The circular process is implemented continuously until the user closes the graphic interface, and the program skips to the End segment to execute close action.

IV. Experimental Results

This experiment attempts to replace the traditional TV System by digital display, and transmit video information via network to remote users, so as to make general traditional TV box more valuable.

A. IPTV Experiment Environment Construction

First, the program development PC side is connected to the TMS320DM6437 EVM by USB transmission line. The PC side is installed with development software Code Composer Studio 3.3 [9]. The DSP execution file (.out) is generated from the composed program by Compiler and Link, finally written in TMS320DM6437 EVM for execution via USB. When the file is executed successfully, the required interface device can be added in the development platform for experiment, as shown in Fig. 8.

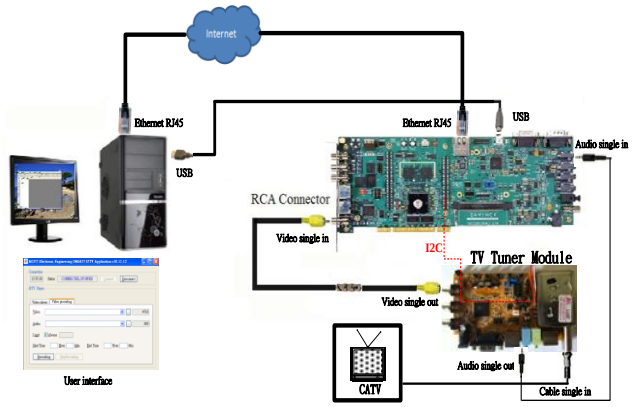

Fig. 8. IPTV platform experiment architecture

Afterwards, the CATV RF signal (Cable line) is connected to the TV tuner module, and then connected to I2C pins of Tuner TNF5535MF and DSP DM6437. When the client-side selects a channel, the DM6437 receives command through the network, and transmits the channel parameter to the Tuner TNF5535MF. The tuner filters the CATV RF signal as soon as it receives the channel parameter to filter unwanted channels. Finally, the demodulation is implemented to divide the video CVBS and audio AF signals of correct channel into two, and is exported to the AV terminal and AUX terminal of circuit. Finally, the video signal is sent via connecting wire to the video single in and audio single in of TMS320DM6437 development board.

When the analog video signal enters the development board, the signal is converted by TVP5146M2 and AIC33 chips into digital data and transferred to DSP DM6437. DM6437 receives the digital video data and then executes H.264 and AAC Encode calculation. Finally, the coded H.264 and AAC data are transferred through network to client-side, which decodes and plays the H.264 and AAC video data as soon as it receives them from the Server-Side, so as to validate the experimental results.

B. Server-Side System

The server-side can carry out two tasks at the same time, not only providing near end video output, but also transmitting it via network interface to remote user to receive and use. Thus, a complete system architecture is finished.

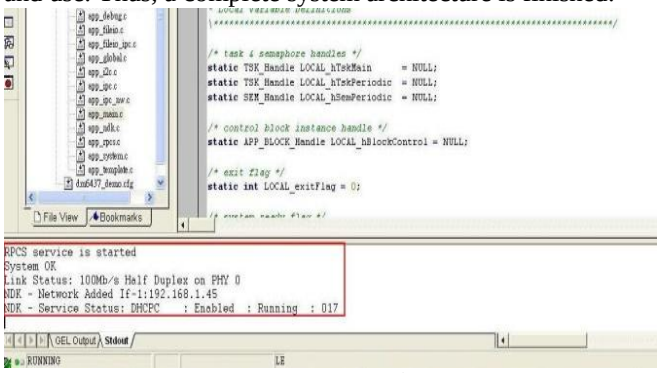

Fig. 9. EVM obtains IP address and waits for client-side connection.

When the development board is in action, the network service is connected actively, waiting for connection to be established at the client-side, as shown in Fig. 9. When the connection is established successfully, the video coding and transmission service begin to work after the client-side transfers parameter values to the server-side system.

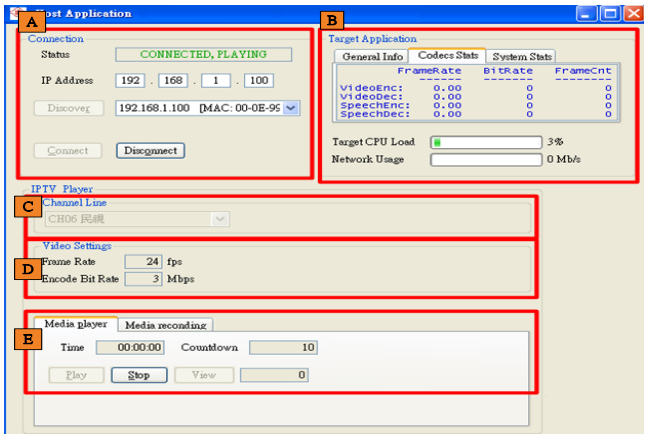

Fig. 10. Graphic user interface

C. Client-Side System

The users' operation habit and intelligent application function are considered in the client-side system development, so the users can connect the server-side easily via this interface to receive the H.264 and AAC digital video information transmitted from the server-side. It can be decoded and played immediately, as shown in Fig. 10. The functions are briefly described according to the symbols A-E in the figure.

1) Connection: establish the server-side connection. First, the server-side IP address must be entered to establish the network connection in between. If the network structure of server-side and client-side is LAN, press the Discover button, and the system will send out UDP broadcast packets via Port 44000. It then waits for server-side response, so as to capture the server-side network IP address. Finally, after pressing the Connect button, the system sends TCP packets via Port 41000 to the server-side for three-way handshake communication protocol, thus, the connection establishment for the client-side and server-side is completed formally.
2) Target application: real-time data detection of server-side, including Frame Rate of H.264 and AAC Encode, Bit Rate, DSP-DM6437 Loading, network flow data between server-side and client-side.
3) Channel line: Channel selection. When the user selects channel on the pull-down menu interface, the system informs server-side via network, and the server-side transmits the channel information via I^2C protocol to the TV tuner module for channel selection. Finally, the analog video of the selected channel is transmitted to the server-side for digital video compression, and transmitted to the client-side via network. Thus, a complete transmission loop system for real-time channel selection is formed.
4) Video settings: H.26 Constant bitrate (CBR). The frame rate and bitrate of H.264 can be fixed by this interface.
5) Media player: Video play control. The interface contains play, stop, view, playing time, video transmission statistics (unit: bytes), transmission rate, play butter counting backwards. When the user presses the Play button, the system transmits various parameters via RPC to the server-side system. The coding and transmission services begin to work when the server-side receives the information.

When the video is transmitted via network to the client-side smoothly, the system buffers 10 sec, and then the system executes the decoding video player automatically to decode and play H.264 and AAC video data, so as to validate the experimental results, as shown in Fig. 11 .

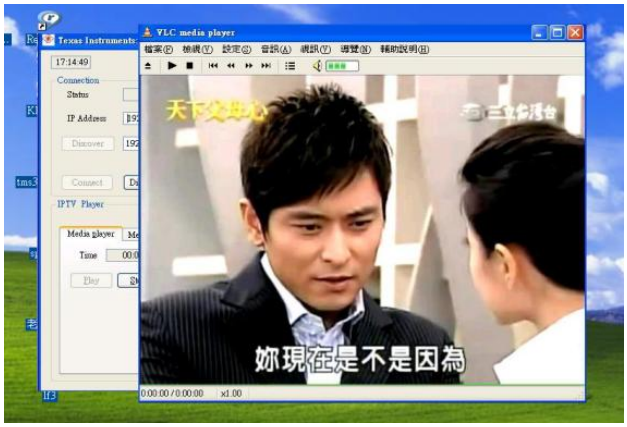

Fig. 11. Video decoding experiment results

D. Network bandwidth Controls Video Quality

The server-side transfers the digital video to the client-side through network. If the network bandwidth on one side is insufficient, the video play will be unsmooth and delayed, thus influencing the picture quality. In order to solve the above problem, this experiment obtains the available network bandwidth between the server and the client by detecting the video flow rate received by the client-side, and adjusts the H.264 bitrate dynamically to meet the smoothness of video play.

The bandwidths for transmitting different levels of H.264 bitrate are shown in Table III. As seen, a higher H.264 bitrate level indicates higher bandwidth utilization rate. If the bandwidth for the server-side transmitting digital video to the client-side is greater than the network usage average in the table, the video play will be unsmooth.

TABLE III: H.264 AND AAC NETWORK TRANSMISSION BANDWIDTH UTILIZATION RATE

H.264 Bitrate	AAC Bitrate	Network Usage Average
3M bps	44kbps	2.7M bps
2.5M bps	44kbps	2.3M bps
2M bps	44kbps	1.9M bps
1.5M bps	44kbps	1.4M bps
1M bps	44kbps	1M bps
0.5M bps	44kbps	0.5M bps

In order to ensure smooth of playing, the network bandwidth between the server-side and client-side must be detected at any time, the video bitrate should be corrected immediately to reduce the video quality if the bandwidth is insufficient, so as to solve unsmooth playing.

E. Experimental Structure

The line bandwidth of server-side and packet generating

server is fixed at 100M bps, and the client-side line bandwidth is fixed at 10M bps, as shown in Fig. 12.

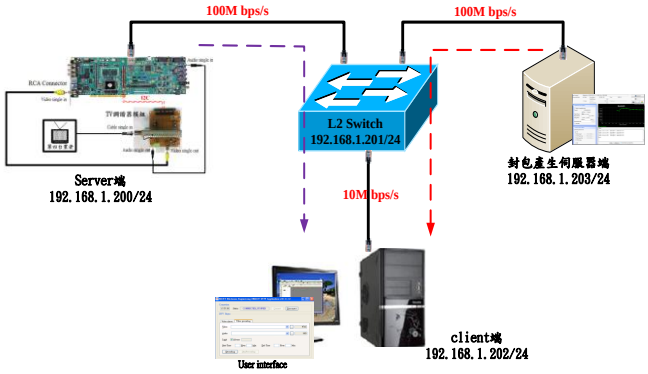

Fig. 12. Pressure test architecture

The packet generating server uses UDP unreliable data stream for bandwidth pressure test, and statistical analysis is carried out by testing the pressure of 6 levels of H.264 bitrate transmission, as shown in Table IV.

TABLE IV: PRESSURE TEST RESULTS

UDP Packets generated	0M	7M	8M	8.5M	9M	9.5M
H.264 3M Bitrate Network Usage /fps	2.6M/ 28fps	2M/20 fps	1.6M/ 15 fps	1.M/ 10fps	0.6M/ 5fps	0.2M/ 2fps
H.264 2M Bitrate Network Usage /fps	1.9M/ 29fps	1.8M/ 27fps	1.4M/ 21fps	1M/15 fps	0.5M/ 7fps	0.2M/ 2fps
H.264 1.5M Bitrate Network Usage/fps	1.3M/ 30fps	1.3M/ 27fps	1.2M/ 22fps	1M/19 fps	0.5M/ 10fps	0.2M/ 4fps
H.264 1M Bitrate Network Usage /fps	0.9M/ 30fps	0.9M/ 30fps	0.9M/ 29fps	0.9M/ 25fps	0.5M/ 15fps	0.2M/ 5fps
H.264 0.5M Bitrate Network Usage/fps	0.4M/ 30fps	0.4M/ 30fps	0.4M/ 30fps	0.4M/ 28fps	0.3M/ 21fps	0.1M/ 7fps

According to Table IV, when the server-side transmits fixed H.264 bitrate video to the client-side, and the bandwidth on both sides is sufficient, H.264 fps has 28~30fps smooth performance. When the packet generating server presses the bandwidth on one side, take H.264 bitrate 3Mbps as an example, the network transmission requires more than 2.6Mbbps bandwidth for smooth playing. When the packet generating server presses 8Mbps UDP packets on the client-side, the ideal bandwidth on the client-side will decrease from 10Mbps to 2Mbps. It is lower than the 2.6Mbps bandwidth requirement. However, in order to meet the picture quality condition of H.264 bitrate 3Mbps, TI H.264 algorithm will reduce fps to meet the situation of only 2Mbps bandwidth, but the picture will mismatch the audio during play, so that the play quality is reduced. In order to solve the above problem, this experiment implements H.264 bitrate dynamic adjustment algorithm immediately when the H.264 fps decrease resulted from insufficient bandwidth is detected, to adjust the H.264 bitrate 3Mbps to 2Mbps to correct the smoothness of video play, as shown in Fig. 13.

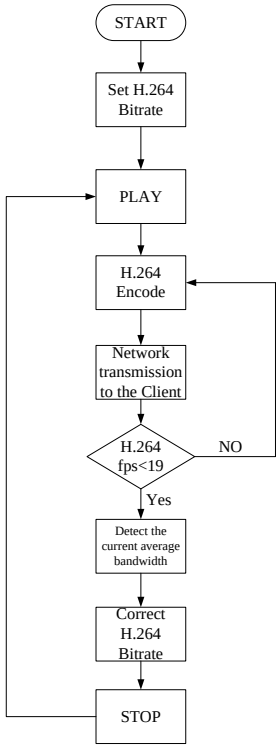

Fig. 13. Flow chart of network bandwidth controlling video quality

F. Efficiency Load Analysis and Discussion

When the overall development process is completed, the efficiency of this system architecture must be tested and evaluated for future evaluation of practical application. The experimental data are shown in the following table. The video adopts D1 (720x480) definition, and the audio is coded by AAC low complexity profile fixed bitrate 44kbps.

TABLE V: H.264 AND AAC CODING LOAD

H.264 Bitrate/fps	AAC Bitrate	DM6437 average Loading (H.264+AAC Encode)
3Mbps/30	44kbps	94%(88%+6%)
2Mbps/30	44kbps	88%(82%+6%)
1.5Mbps/30	44kbps	83%(76%+6%)
1Mbps/30	44kbps	69%(63%+6%)
0.5Mbps/30	44kbps	61%(55%+6%)

According to Table V, the capacity of H.264 coding is very large for DSP DM6437, the average is about 55% from bitrate 0.5Mbps, and increases to about 88% at bitrate 3Mbps. If AAC coding and network transmission are executed at the same time, the capacity of DSP DM6437 will be more than 90%. It is thus likely to cause overheating of DM6437 processor and system crash.

Therefore, when the DSP DM6437 capacity is considered, the difference in the picture quality higher than H.264 bitrate 1.5Mbps cannot be detected by naked eye. The average load of DSP DM6437 is about 76%, which is still within the acceptable range of DM6437 load. The network transmission bandwidth is only 1.3Mbps. Therefore, DM6437 is used for

video transmission service in this experiment, the recommended H.264 bitrate is below 1.5Mbps for video coding operation, so that this experimental system can work more stably.

V. Conclusions and Future Studies

This work implemented TI TMS320DM6437 digital signal processing platform in network video transmission successfully. The H.264 video compression coding wrapped by TI conforming to xDAIS (eXpress DSP Algorithm Interoperability Standard) and Advance Audio Coding (AAC) were implemented in digital signal processor. The TI DSP Network Developer's Kit (NDK) [10] was integrated with TV tuner module to implement network sockets connected transmission between the server-side and client-side, as well as TV channel selection control. The client-side receives the real-time multimedia video from DSP server-side successfully via Javascript language program client-side interface, and decodes and plays it. Finally, the network flow detection controlled video quality and efficiency load analysis are discussed, it is the major finding of this paper.

This work is still at the experimental stage at present. It can be the basis of subsequent research and development, thus creating new technology in the future. In order to become a practical product, more factors must be considered, such as using RTP/RTCP network protocol, webpage video browsing mode, and supporting more video algorithms, reducing DSP calculation load, etc. All of which will be problems in the future; however, the IPTV system is integrated with more diversified applications and more user-friendly functions as a result.

Whether the IPTV technology will be popular in the next two to three years, or replaced by other more advanced technologies, the spirit of transferring video information via networks will not change. We will keep developing new techniques and functions in the future to further complete and enrich IPTV.

References

[1] I. E. G. Richardson, H.264 and MPEG-4 Video Compression: *Video Coding for Next-generation Multimedia*, John Wiley & Sons, Ltd, UK, pp. 160-206, 2003.

[2] Texas Instruments Inc., "AAC Encoder on C64x+," Literature Number SPRUEI4A, January 2008.

[3] Texas Instruments, Inc. "DaVinci Technology Background and Specifications," Literature Number SPRT401A, January 2007

[4] Texas Instruments Inc., "TMS320DM6437 Digital Media Processor," Literature Number SPRS345D, June 2008.

[5] Spectrum Digital, Inc., "DM6437 EVM Technical Reference," Manual No. 509105-0001 Rev. C, Dec 2006.

[6] Shenzhen Tena 1 Electronics Co., Ltd. "TNF5535MF Product Specification, " 2004.

[7] Texas Instruments, Inc."xDAIS-DM (Digital Media) User Guide," Literature Number SPRUEC8B, January 2007

[8] B. Caldwell, *Davinci DM6437 DVDP Reference App / Demo Overview*, March 2007.

[9] Texas Instruments, Inc. "Code Composer Studio Development Tools v3.3 Getting Started Guide," Literature Number SPRU509H, May 2008.

[10] Texas Instruments Inc., "TMS320CG000 Network Developer Kit(NDK) " User Guide," Literature Number SPRU523C,January 2007.

Yen-Jen Chen was born in Taiwan, R.O.C. He received the B.S. and Ph.D. degrees in computer science and information engineering in 1989 from Tatung Institute of Technology and in 2000 from National Chiao Tung University (NCTU), Taiwan, respectively. From October 2000 to July 2001, he was an Assistant Professor under the collaboration project of NCTU and Cisco Systems Inc. for developing VoIP Testing Technology. From. August 2001 to August 2002, he was a network design consultant in Unisys Corporation, Taiwan. Since August 2002, he is an Assistant Professor in the Department of Electronic Engineering, Ming Chi University of Technology, Taiwan. His research interests include traffic engineering and quality of service in computer networks, network management and system integration, and cloud virtualization

Chih-Chung Wang received his B.S. in Electronic Engineering from Ming Chi University of Technology, Taiwan, R.O.C., in 2009. He is a postgraduate at Ming Chi University of Technology, Taiwan, and his research interests include network , embedded system, and web application development.

Jr-Jen Huang received the B.S.E.E. degree from Chung Yuan Christian University, Taiwan, R.O.C., in 1988, and the M.S.E.E. degree from the State University of New York, Binghamton, in 1991, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN in 1999. Dr. Huang is currently an assistant professor at Ming Chi University of Technology, Taiwan and his research interests include nonlinear signal and image processing techniques, estimation under the mean absolute error criterion, perceptual error measures, stack filters and applications of LBS system.

Dr. Huang received Outstanding Teacher Awards from Ming Chi University of Technology class of 2009 and Taiwan Private School Society in 2010. Dr. Huang has been serving as the director of Library and Information Services of Ming Chi University of Technology, Taiwan since 2010