

<http://ansinet.com/itj>

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSI*net*

Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Embedded Remote Monitoring System Based on Internet

Qingnan Fan

College of Software, Shandong University, 250101 Jinan, China

Abstract: To overcome drawbacks of PC monitor server and C/S mode in the traditional monitoring scheme, this paper designs a new embedded server system for remote monitoring solutions based on B/S mode. This system is implemented by employing South Korea's Samsung S3C2410 microprocessor as hardware core and embedded web server as software core. It combines MPEG-4 video image compression algorithm, BOA embedded web server and CGI web programming technology to realize the function of monitoring video terminals' field data wirelessly in the remote Web client. This study uses modular structure and has the advantages of good stability, independence and flexibility, with broad application prospects.

Key words: S3C2410, CGI, MPEG-4, remote monitoring solution, embedded server system

INTRODUCTION

The remote monitoring system is mainly used for achieving remote field data acquisition, processing and real-time monitoring functions. Traditional remote monitoring system uses analog video surveillance system. The morphology of information flow in the analog video surveillance system is analog video signals. The network structure of the system is a single-function, one-way, lumped-way information collection network. Due to his limited transmission distance, high bandwidth, weak ability to allocate bandwidth flexibly, complex operations, this system is limited by a lot (Cager, 2006; Koskelo *et al.*, 1999).

PC-based digital video surveillance system is mainly dependent on the mixed analog-digital or fully digital video transmission and processing methods (Tournas and Georgopoulos, 2001; Boochs *et al.*, 2002). This system will convert the analog audio and video signal collected by the microphone and camera to digital signals, then compress the digital signals and transmit them to the PC monitoring terminal through the network. Compared to analog video surveillance, PC-based multimedia surveillance system has characteristics of far transmission distance, good image quality and low data storage costs, but the stability of the system is not good, the reliability is not high and the flexibility is not strong. PC also needs expert management. So this approach is not ideal.

In recent years, with the continuous development of network technology, video transmission and compression technology, video surveillance systems undergo a transition from analog to digital gradually and present the developing trend of network and integration (Bramberger *et al.*, 2004). Embedded Web-based network video surveillance system has received public widespread concern gradually. Web digital video surveillance system

has been developed to become the mainstream of video monitoring system (Bramberger *et al.*, 2006).

This study does research into the remote monitoring system for embedded Internet and builds a remote video surveillance system of embedded B/S mode based on ARM processor and Linux operating system (Yang and Li, 2008). The system integrates image capture, video compression and Web technologies into a very small device that can be connected to LAN and Internet directly to attain the plug-and-play function. It cuts a variety of complex cables to make it's easy to install. The user does not need to install any hardware device to monitor the scene.

SYSTEM DESIGN AND PRINCIPLE

System design: As a whole, this system is composed of three parts. The front-end is USB camera for video capture. The middle part is ARM9 development board transplanted to embedded Linux operating system. Its role is to compress the collected digital image signals with MPEG-4 algorithms (Ebrahimi, 1997). This board has built-in embedded Web server and video streaming server, which transmits videos to the remote user. The third part is the monitoring client composed of a PC with a browser, which is used to browse the remote video and control system platform through the network.

As shown in Fig. 1, the embedded remote video capture system consists of two parts mainly: hardware and software. The hardware part contains USB camera and ARM9 development board; the software part contains system softwares (including Bootloader, embedded Linux operating system and drivers) and applications (including Web server, CGI program, video capture, coding and PTZ control etc.). The software part is the key point of this design.

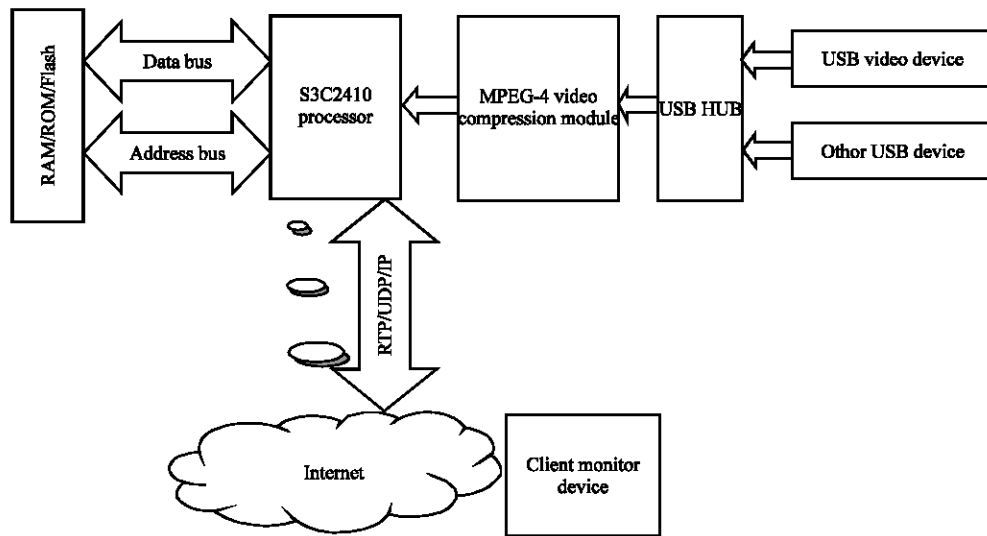


Fig. 1: Overall framework of remote monitoring

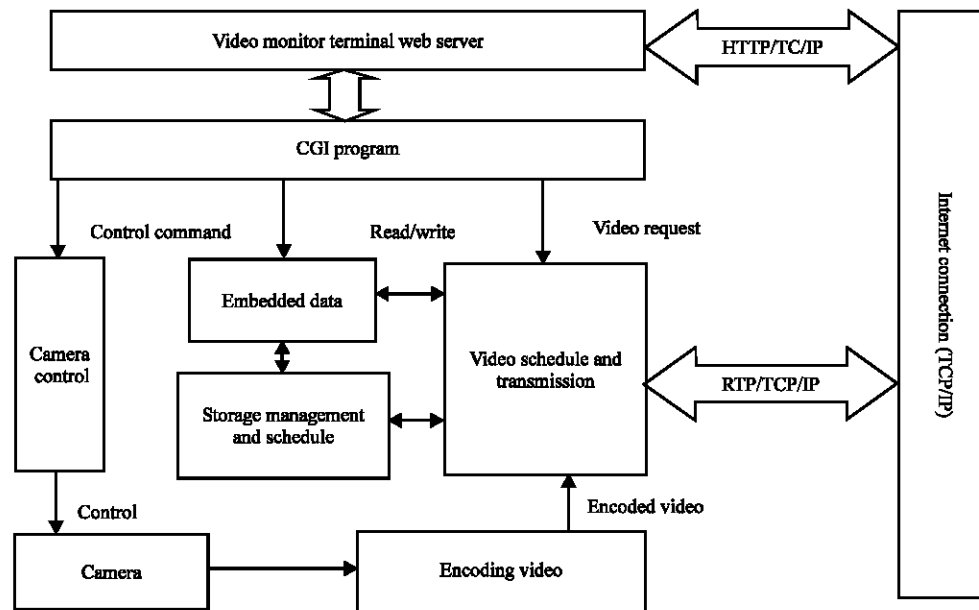


Fig. 2: Software structure of video surveillance system

System working principle and process: The application software structure of video surveillance terminal is shown in Fig. 2. It is mainly composed by WEB server, CGI program, embedded database system, video scheduling and transmission module, storage management and scheduling module, camera control module and several other important parts.

The system collects images by the camera, captures on-site images at a high speed. Then the captured images are transferred to the S3C2410 processor for processing

and compressing through the USB bus and saved as JPEG format files. Next the program calls the encoder to synthesize multiple JPEG format images for some AVI video stream and implements video playback, which has a wide range of applications in the field of security monitoring, incident identification, vehicle anti-theft. Finally, the system can also upload the saved pictures and video to server through the Ethernet port or UART port in order to monitor by Internet.

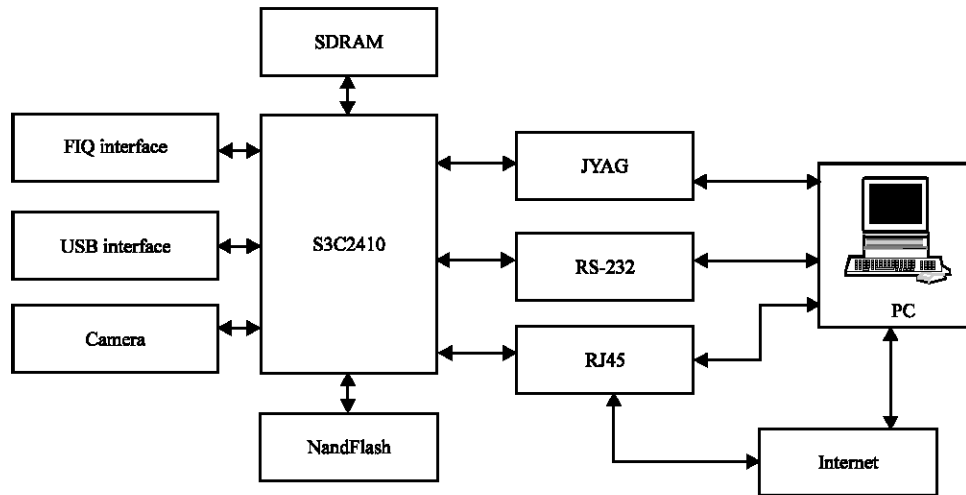


Fig. 3: Software structure of video surveillance system

SYSTEM HARDWARE DESIGN

Zhou and Chen (2004) introduced the working mechanism and internal structure of Samsung's S3C2410 chip based on ARM920T kernel. (Song *et al.*, 2008) put forward a transplanting solution of ARM Linux on the S3C2410 embedded platform system and they introduced the structure of hardware platform and operation process. This system draws on their previous practices and develops its own unique pattern.

The system hardware is made up of USB camera, S3C2410 embedded processor, NandFlash, the JTAG, RS232, RJ45 interface and other components. Video signals are captured by USB camera, compressed by MPEG-4 encoding algorithm and transmitted to the network via RJ45 interface. FIQ is temperature, infrared sensor interface. It can be used for home security. The system is connected to PC through RS232 and RJ45 interface to form a cross-compiler debugging environment. Video acquisition and compression module is implemented by software. This system communicates with S3C2410 through JTAG interface on the development board. System hardware structure is shown in Fig. 3.

CPU: The main control chip S3C2410. This chip is a RISC microprocessor based on the ARM920T core produced by SAMSUNG. It inherits memory interface, USB, RS232, RJ45, FIQ, JTAG interface and other hardware resources.

Nand flash: K9F1208 chip. Monolithic storage capacity of the chip is 64M×8bit, operating voltage is 2.7V ~ 3.6V.

USB camera: The CMOS camera sensor (Nixon *et al.*, 1996). The A/D conversion, timing control, signal

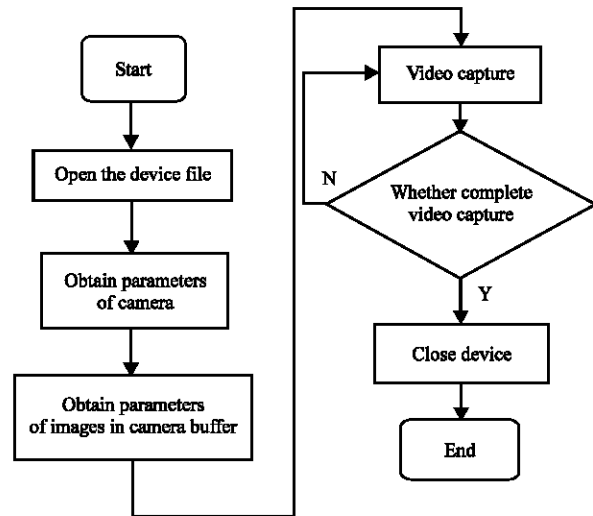


Fig. 4: Specific workflow of video capture program

processing, color-coding, compensation and conversion control modules are integrated on the same chip with high integration, high speed, low power consumption, low price and small size, etc.

SYSTEM SOFTWARE DESIGN

V4L-based video capture: Video acquisition based on Linux is realized by Video4Linux. Video4Linux bus standard is used widely and provides a series of interface functions for application programming of video equipments. The process of Linux video capture is as follows (Fig. 4):

- **Step 1:** Open the device file. It specifies camera device file /dev/video0 and calls open () to open the device file, `Int fd = open ("/dev/video0", O_RDWR)`
- **Step 2:** Get parameters of the camera and execute function `Ioctl (_fd, VIDIOCGCAP, and capability)` to read information about the camera in the structure `struct video_capability` (device name, maximum and minimum resolution, signal source)
- **Step 3:** Set image parameters in the camera buffer
- **Step 4:** Video capture. Video acquisition has two methods: `mmap ()` memory mapping mode and reading directly mode. The `mmap ()` system call shares memory between processes by mapping a regular file. After the regular file is mapped into the address space of process, the process can access files like common memory and does not have to call `read ()`, `write ()` and other functions. An obvious benefit of communication with shared memory is high efficiency, because process can read and write memory directly without need for a copy of any data. The function to intercept video in `mmap ()` mode is `(vd-> fd, VIDIOCMCAPTURE and (vd-> mmap))`. If the function succeeds, an interception of a frame begins. It's non-blocking and whether the interception is completed is judged by `VIDIOCSYNC`. And then, we need to call `(ioctl(vd->fd,VIDIOCSYNC and frame))` to wait for the end of an interception. We can start another `VIDIOCMCAPTURE` when it's completed successfully
- **Step 5:** Clean up memory and turn off the video equipment

MPEG-4 compression of video image: Video data compression module is to compress and encode collected video by MPEG-4 compression standard, generate MPEG-4 video data stream and transmit the stream to requesting monitoring clients via network by video data transmission module. This system uses open-source and efficient multimedia codec software `xvidocre` as the core of MPEG-4 video compression algorithm. Steps of compressing and encoding videos by XVID encoder are as follows:

- **Step 1:** Initialization. Create two most critical structures: information structure of coded frame (`xvid_enc_frame_t`) and status information structure of coded frame (`xvid_enc_stats_t`), used for incoming parameters and calculating coding results, respectively
- **Step 2:** Read the first frame of image. It calls the function `read_yuvdata ()` to read the information of first frame from original file buffer YUV and transmits

the corresponding arguments of image to defined structure `xvid_enc_frame_t` and `xvid_enc_stats_t`

- **Step 3:** Intra-frame coding. The system calls the function `CodeIntraMB ()` to set the coding mode to Intra and all sports-related variable to 0. If the value of differential quantization is not 0, we should set it to intra-Q. Then it calls the encoding function `static int FrameCodeI (Encoder *pEnc, Bitstream *bs)` for intra-frame encoding
- **Step 4:** Exchange a reconstructed frame for a reference frame. It takes a picture from frame queue as a coded frame, i.e., the current frame and initializes the frame (same with the first step)
- **Step 5:** Call the function `xvid_encode (enc_handle, XVID_ENC_ENCODE, and xvid_enc_frame, and xvid_enc_stats)` to encode frame image
- **Step 6:** Encode frame according to the coding mode determined by the fifth step. If it's intra-coding mode, turn to the third step; if it is inter-frame coding mode, call P-frame encoding function

BOA embedded Web server's transplantation: In the embedded remote monitoring system, in order to enable the remote hosts get image and video data via the Internet, we need to transplant a Web server that support script and CGI technology in the embedded system. The performance of the Web server determines the overall performance of the system. The typical embedded Web server includes BOA and `Thttpd`. The BOA server and `Thttpd` server both support certification and CGI technology. Clients can manage and monitor the embedded devices via IE browser. The system chooses open-source BOA, whose executable code size is only about 60KB and that is a very compact Web server. Moreover, it is a single-job Web server and can only meet the user's service request one by one, rather than the Linux Fork function creates a new process to generate concurrent requests. But it can create a process for the CGI program, whose performance of speed and safety is relatively good.

Process to create a BOA server:

- **Step 1:** Download the BOA server source code file: `BOA-0.94.13.tar.gz` and unzip it in the directory `/BOA/src/`
- **Step 2:** Compile the BOA. Execute `/BOA/src/configure` when transplanting BOA and generate a Makefile. In the Makefile, change "`CC = gcc`" to "`CC = arm-linux-gcc`" and "`CPP = gcc -E`" to "`CC = arm-linux-gcc-E`", which is the arm-linux-gcc version. Then it modifies the compiler to

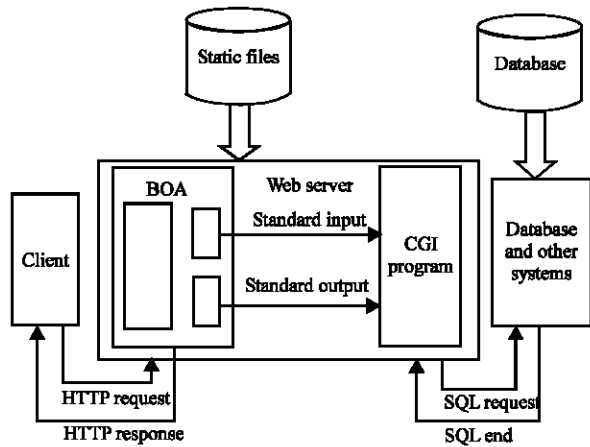


Fig. 5: CGI work principle diagram

cross-compiler for the Web server that supports the S3C2410 platform. Subsequently we need to modify the BOA's root file directory in defines.h: #define SERVER_ROOT "/etc/BOA" and execute the make command to compile the BOA source code, which will generate the BOA's executable file. Modify the src/compat.h, src/log.c, src/BOA.c etc

- **Step 3:** Configure BOA server. To make BOA run in the corresponding embedded platform, we need to configure its operating environment, parameters, etc. The main part of this step is to modify the storage path in the file BOA.conf
- **Step 4:** Copy the BOA and BOA.conf files to the corresponding directory of the embedded systems. Then we can access BOA Web Server

The embedded Web server provides network access and information services. The system is based on TCP/IP protocol, HTTP protocol. It can call CGI programs with data request and control functions to display information dynamically in a browser. It achieves the purpose of remote monitoring.

The transplanting steps of embedded database SQLite is similar as above.

CGI Web interactive programming: The CGI technology (Common Gateway Interface) supports dynamic refresh, conversion and display of Web data.

CGI is a program running on the Web server. Like other types of program, it must be in accordance with the CGI standard format. CGI programs run by the client browser input trigger. Its task is to execute instructions, convert the required data to environment variables and return the processing results. The Web server and CGI

program communicates by four ways: environment variables, command line, standard input and standard output.

CGI basic working process: According to the request of the client browser, CGI programs call other applications for processing by executing the BOA server's commands. Finally, the results return back to the client browser to display in the form of HTTP. The CGI workflow diagram is shown in Fig. 5.

CGI is an interface to run an external program in the Web server. CGI programs make web pages interactive. The most important role of CGI is to provide a number of functions that HTML cannot realize.

CONCLUSION

This study is about an embedded remote monitoring system based on S3C2410 microprocessor and taking advantage of Linux open source code. The study analyses functions and composition structure of the wireless video surveillance system, studies the characteristics of hardware components and software implementation of the system, determines the modular way to develop this system, such as video capture module, video compression module, video transmission module and so on. The independence of each module enhances robustness and flexibility of the system. When we need to replace a module and other modules do not need to make big changes, it is conducive to the upgrading of the system. Embedded Web remote monitoring system has the advantages of low cost, small size, good stability and reliability, easy installation and strong practicability. It is becoming the main force of industry monitoring system.

REFERENCES

- Boochs, F., S. Eckhardt and B. Fischer, 2002. A PC-based stereoscopic measurement system for the generation of digital object models. Bar Int. Ser., 1016: 371-378.
- Bramberger, M., A. Doblander, A. Maier, B. Rinner and H. Schwabach, 2006. Distributed embedded smart cameras for surveillance applications. Computer, 39: 68-75.
- Bramberger, M., J. Brunner, B. Rinner and H. Schwabach, 2004. Real-time video analysis on an embedded smart camera for traffic surveillance. Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium, May 25-28, 2004, USA., pp: 174-181.

- Cager, Y., 2006. Smart video surveillance: Digital technology vs. tradition analog systems. *Adv. Imaging*, 21: 12-16.
- Ebrahimi, T., 1997. MPEG-4 video verification model: A video encoding/decoding algorithm based on content representation. *Signal Processing: Image Communi.*, 9: 367-384.
- Koskelo, M.J., I.J. Koskelo and B. Sielaff, 1999. Comparison of analog and digital signal processing systems using pulsers. *Nuclear Instruments Methods Phys. Res. Section A: Accelerators Spectrometers Detectors Assoc. Equipm* 422: 373-378.
- Nixon, R.H., S.E. Kemeny, B. Pain, C.O. Staller and E.R. Fossum, 1996. 256 x 256 CMOS active pixel sensor camera-on-a-chip. *IEEE J. Solid-State Circ.*, 31: 2046-2050.
- Song, K., L.P. Yan, L. Gan and X.S. Huang, 2008. Porting ARM Linux to S3C2410. *Comput. Eng. Design*, 29: 4138-4140.
- Tournas, L. and A. Georgopoulos, 2001. Stereoscopic video imaging using a low-cost PC-based system. *Proc. Spie Int. Soc. Optical Eng.*, 4309: 312-317.
- Yang, N. and F. Li, 2008. Design and implementation of surveillance system for video based on B/S. *Comput. Eng. Design*, 21: 5576-5579.
- Zhou, W. and M. Chen, 2004. The ARM training-kit based on S3C2410. *Electron. Technol.*, 7: 4-7.