Indian Journal of Science and Technology, *Vol 9(20), DOI: 10.17485/ijst/2016/v9i20/94670, May 2016*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Study on Event Data Recorder RTOS Platform Connected to the Wireless LAN Providing Progressive Download Method

Seung-hoon Lee and Eui-Seok Nahm*

Department of Ubiquitous IT, Far East University, Republic of Korea;
shlee@kdu.ac.kr, nahmes@kdu.ac.kr

Abstract

Background/Objectives: In these days, the car black box is to be fast in boot-up time with low power consumption from battery. The aim is to develop a product with minimum cost yet in support with Wi-Fi communication with fast boot-up time. **Methods/Statistical Analysis:** To bring down the cost, we use common Wi-Fi dongle and the low capacity SPI (Serial Peripheral Interface); NOR flash memory is adopted based on RTOS supporting HTTP streaming for the fast boot-up time and lesser power consumption. Using RTOS technology, we developed an MP4 encoder/decoder/muxer. **Findings:** Up to ten seconds of booting speed, minimum memory and the advantage of operating at lower power consumption was achieved than a general-purpose Linux; Utilizing Wi-Fi dongle provide smart phone streaming functionality and lower product price per unit. The developed black box is supporting full HD 30 fps and 2 channels. By field testing, it has 10% lower power consuming and faster boot-up time. The cost can be reduced by using common chips comparing with high specific products. **Improvements:** This MP4 component may further be enhanced for cost effective

Keywords: Black Box, Event Data Recorder, Progressive Download, RTOS, Wireless LAN

1. Introduction

A vehicle video recording and storage devices are effective in the areas such as vehicle speed recording, location identification and accident prevention. These are of great demand in the automobile parts markets. Various methods of Wi-Fi based image information transmission to the smart phones connected to an EDR have been suggested. The latest product is the Wi-Fi supportive, but by default, the chip companies only support Linux platforms because they supply mobile AP (Application Processor). These chip companies prefer to reduce license price by leveraging open source and platform development with small amount of investment[1].

However, the vehicle image storage device needs to accelerate boot-time for recording at an early stage. This is to consume the least amount of processing power due to the limits of battery capacity of the car. Requirement of minimizing the use of memory is for the price competitiveness of products[2]. We need to make products with minimum cost yet in support with Wi-Fi communication for its stability and reliability. In this paper, protocol design and USB Driver port and embedded TCP/IP stack port, DHCP server and the Web were implemented to provide video stored information in the vehicle's device to the smart phones by Progressive download method. This is to solve the Real-Time Operating System (RTOS) linked to the encoder and to develop the muxer working together with streaming service[3].

Author for correspondence

2. Event Data Recorder RTOS Platform that offers Streaming on a Smart phone

Key technical elements necessary for implementation of the Block Box are technology of sensing the potential accidents, information storage technology and information management technology and information analysis techniques[4].

As EDR was gradually expanding, from one channel of VGA products to HD-2 channel products, then again two-channel Full HD products have evolved. Nowadays such as LCD feature containing high-end products are being sold[5].

Boot time, processing power consumption of the vehicle image at the existing vehicle Block Box and for the product price competitiveness to improve with its memory usage to minimize, mobile AP applied with RTOS, with automotive MP4 encoder/decoder/Muxer need to be developed. Furthermore, GPS and G-Sensor algorithms, along with the User Interface of the storage under the RTOS-based have been implemented. They have booting speed up to ten seconds, minimum memory and the advantage of operation at reduced performance than Linux. The final goal was set to develop video recording device RTOS platform to provide streaming capability to the smart phones with a low product cost per unit by using the multi-purpose Wi-Fi dongle.

The use of TCP/IP, the Web and a DHCP server porting by combining it with the USB 2.0 Host interface to support Wi-Fi on the RTOS enable to overcome weakness of the RTOS system implemented with a protocol stack[6]. By using the RTOS-based platform, it accelerates faster boot time and takes less memory. Its advantages go far by consuming low processing power compared to Linux. The main contents of the technical development are shown in the Table 1.

2.1 Video Streaming System

2.1.1 Progressive Download

It is video transmission system widely used in the VOD (Video On Demand) plays content while gradually downloading using a standard Web browser and not on a separate media server though the <video> tag in HTML5 and the video formats are provided with mp4, webm, ogg.

Table 1. Key technology development

Key Technology Development	Technology
Device Driver RTOS running on a general-purpose WiFi USB dongle	USB OTG, WiFi, RTOS
Operating embedded TCP / IP stack port and Web, DHCP Server Implementation on the RTOS	TCP/IP, HTTP, DHCP
Encoder and muxer development by which camera images can be interlocked with a web server to provide streaming service	YUV Format (I420/NV12), H.264 Encoder, MP4
Development of the real-time previewing camera video using smartphone browsers	HTML, AJAX, HTTP Streaming

In this download, one can specify the location of the image to be downloaded directly from the client. The HTTP headers are Content-Ranges and Accept-Ranges. Client will tell the starting position to download via Content-Ranges tag, then, the web server responds in the form as shown below[7]:

2.1.2 libjpeg-turbo

This library supports the same libjpeg which functions as an encoding open source library from YUV format Video images to that of the encoded JPEG images. However, with the use of SIMD (Single Instruction Multiple Data) instructions called ARM NEON, this is an open source library that supports acceleration of the speed faster than the libjpeg by 2-4 times. The way of compilation is as follows:

Use gcc toolchain on Linux.

Use RVCT on Cygwin.

Install Code Sourcery tool chain on Cygwin after you add the <toolchain> / bin folder to your PATH to correct the under part.

-. Modify the configure file to host_cpu = arm and host_os = Linux.

-. Modify config.guess files to UNAME MACHINE = arm and UNAME SYSTEM = Linux.

2.2 Embedded TCP/IP and Server

2.2.1 Embedded TCP/IP stack

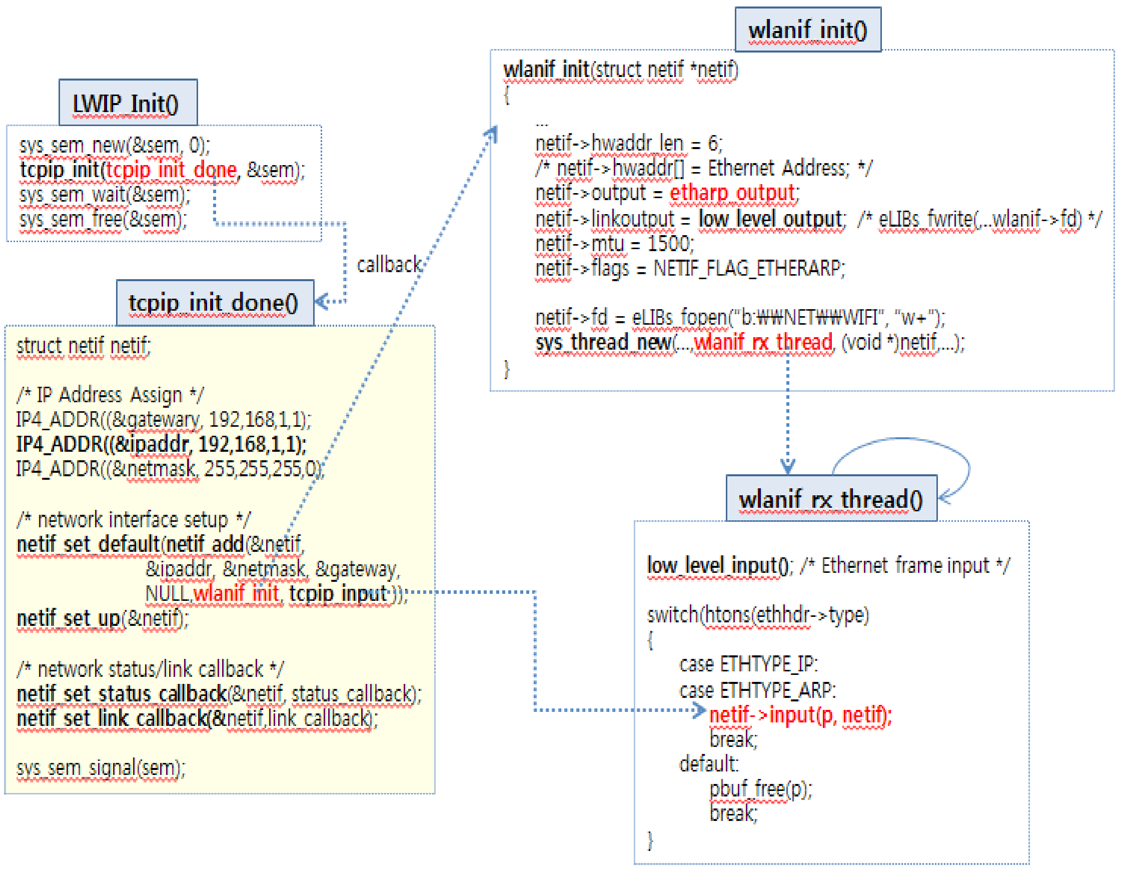
Open source TCP/IP stack

- lwIP(Light-weight IP): In the embedded environment, the stack used by most TCP/ P stack has been ported to this paper and it supports IP, ICMP, TCP/UDP, IGMP, ARP, PPPoS, PPPoE protocols.
- µc/IP (Microcontroller IP): The capacity is often used for small microcontroller TCP/IP stack.
- µIP (Micro IP): Open source TCP/IP stack is used in the OS, Contiki for IoT.
- FNET: HTTP server, DNS server/client and TCP/IP stack that includes a DHCP client, such as together

lwIP stack porting process

- System customized by lwipopt.h.
- Ports/target/include/arch/cc.h typedef, endian - related declarations and debug in the file.
- Declared for mutex, semaphore, mailbox at ports/target/include/arch/sys_arch.h file.
- In the ports/target/sys_arch.c file, mutex, semaphore, mailbox, timer, thread related API for porting.
- Ports/target/netif/netif.c network interface porting on the file.

Figure 1 shows the lwIP stack Initialization. When the tcpip_init() is called, the call back functiontcp_init_done() is enrolled after finishing the inner initializing. The function tcp_init_done() use IP and network interface as arguments of netif_add() for setting of TCP/IP. And when TCP/IP applications send the packets by function send(), low_leve_output() is uesd through etharp_output().



```
sys_sem_new(&sem, 0);
tcpip_init(tcpip_init_done, &sem);
sys_sem_wait(&sem);
sys_sem_free(&sem);
```

```
struct netif netif;

/* IP Address Assign */
IP4_ADDR((&gatewary, 192,168,1,1);
IP4_ADDR((&ipaddr, 192,168,1,1);
IP4_ADDR((&netmask, 255,255,255,0);

/* network interface setup */
netif_set_default(netif_add(&netif,
                  &ipaddr, &netmask, &gateway,
                  NULL,wlanif_init, tcpip_input));
netif_set_up(&netif);

/* network status/link callback */
netif_set_status_callback(&netif, status_callback);
netif_set_link_callback(&netif,link_callback);

sys_sem_signal(sem);
```

```
wlanif_init(struct netif *netif)
{
    ...
    netif->hwaddr_len = 6;
    /* netif->hwaddr[] = Ethernet Address; */
    netif->output = etharp_output;
    netif->linkoutput = low_level_output;  /* eLIBs_fwrite(,..wlanif->fd) */
    netif->mtu = 1500;
    netif->flags = NETIF_FLAG_ETHERARP;

    netif->fd = eLIBs_fopen("b:₩₩NET₩₩WIFI", "w+");
    sys_thread_new(...,wlanif_rx_thread, (void *)netif,...);
}
```

```
low_level_input(); /* Ethernet frame input */

switch(htons(ethhdr->type)
{
    case ETHTYPE_IP:
    case ETHTYPE_ARP:
        netif->input(p, netif);
        break;
    default:
        pbuf_free(p);
        break;
}
```

Figure 1. lwIP stack initialization.

2.2.2 Embedded TCP/IP Stack

HTTP 1.0 is implemented by the ability to send and receive image data and the image browsers. Web server functionality added in the HTTP 1.1 is as follows:

- Host header transaction.
- Allow absolute path URL.
- Chunked mode - Data using the Transfer-Encoding header instead of using the Content-Length header in the server by breaking into several pieces sent from the server to the form below to send away. End of data is divided into '0'\r\n\r\n.

- Keep-alive function added into Connection header.
- Date header.
- If-Modified-Since and If-Unmodified-Since header added.
- 100 Continue added.
- GET and HEAD method support required.
- Cache-Control header added.

2.2.3 DHCP Server

Figure 2 shows the DHCP operation flowchart. The IP address assigned by DHCP is not for the permanent use. The IP address assigned to the client does not have permanent lease time, instead, but the specified rental period. During which, the client is to use only the assigned address. If, even after the lease time ends, you want to continue to use the IP address, you should request for a re-leaseing[8].

Client is assigned to an IP address from the DHCP server to use the IP address. The address can be used immediately after receiving a separate approval of the Request packet from the DHCP server. The DHCP server sends an IP address, lease time, subnet mask, default gateway address, and DNS together with the Ack message.

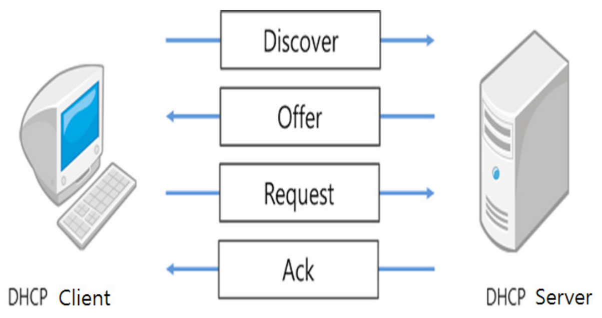


Figure 2. DHCP operation flowchart.

2.3 Realtek WiFi USB Dongle

2.3.1 802.11 Wireless LAN

Wi-Fi is short-range wireless communication technology evolved from various forms of technology with the rapid expansion of the smart phone market since 2010. Initially through 11 Mbps, 802.11b and 802.11a /g of 54 Mbps speed, nowadays it is popularized by 802.11n of up to 600 Mbps speed supporting both 2.4 GHz and 5 GHz. Nowadays it is evolving to IEEE 802.11ac operating in the 5 GHz at Gigabit speed and IEEE 802.11ad type operating at 60 GHz[9].

With 802.11n, compared to 802.11a/g, four more sub-carriers can be used by supporting 40 MHz bandwidth using non 400 ns shorter GI (Guard Interval) and 20 MHz. Therefore, if you use both of these functions, it supports the basic 150 Mbps speed.

In the existing 802.11a/b/g system, MIMO (Multi Input Multi Output) is not supported. Therefore, at one time, only one Tx and Rx antenna may be used. As a consequence, out of the two antennas, one may depend on and is used for the diversity purpose[10].

Wi-Fi is divided into Infrastructure mode and Ad Hoc mode, according to the configurations. Infrastructure mode means a network of the Station wireless AP (Access Point). Ad Hoc mode refers to a mode of operation as Wi-Fi Direct or p2p system such as Wi-Fi TDLS[11].

Wi-Fi USB interface method used in this paper has the following features:

- Single-Chip IEEE 802.11b/g/n.
- Two Transmit and Two Receive path (2T2R).
- 2x2 MIMO OFDM for 2.4GHz.
- 144.4 Mbps using 20 MHz bandwidth.
- 300 Mbps using 40 MHz bandwidth and 400 ns Short Guard Interval.
- USB 2.0 Interface
- Frame Aggregation for increased MAC efficiency (A-MSDU, A-MPDU).
- Low latency immediate High-Throughput Block Acknowledgement (HT_BA).
- Device Driver compatible with RTL8188CUS (1T1R.)
- IEEE 802.11e (QoS Enhancement), 802.11h TPC (Transmit Power Control), Spectrum Measurement 802.11i (WPA/WPA2), 802.11k (Radio Resource Measurement) support.

2.3.2 LINUX Driver RTOS Porting Conside Rations

- Conversion Tasklet and Workqueue

ISR (Interrupt Service Routine) should be run as quickly as possible because it does not run and is not interrupted in the process context. The worst case is with all interrupts being carried out. To solve this problem, the interrupt processing is processed by dividing the top half and the bottom half. That is within the ISR to handle the time-critical job as you need to respond immediately to the hardware called "top half". For the relatively less important tasks they are to return from ISR and are to execute delays since interrupts enabled them. This is called as "Bottom half". To implement the Bottom Half, the following methods in shown Table 2 are used in Linux[12].

In this paper, we use the AP manufacturer in its own RTOS non-Linux OS. It does not support the ability to delay the work, such as tasklets and work queue. Therefore, it was implemented through the common thread and the thread associated with API is as follows.

__u8 esKRNL_TCreate(__pTHD_t thread, void *arg, __u32 stksize, __u16 prioLevel)

__s8 esKRNL_TDel(__u8 tid);

- Hostapd Implementation role

Hostapd is a user-space daemon program that is responsible for Wireless Router (Access Point) and the authentication server role for the wireless LAN card to use the mac802.11 subsystem of Linux kernel. The program had implemented the IEEE 802.11 AP management capabilities, IEEE 802.1x/WPA/WPA2/EAP, RADIUS authentication server and client functionality.

2.3.3 Problems and Fixes

There has been a problem that does not work properly in USB 2.0 High Speed due to USB OTG Host Driver of RTOS. The cause of the problems was identified as it does not combine the data read from the USB Rx FIFO with the data received via the DMA. This is when USB Short Packet smaller than 512 bytes is received[13,14].

2.4 Overall System Configuration

2.4.1 Software Specifications

- Protocol Schematic

Figure 3 shows the proposed protocol scheme. The procedures are as following:

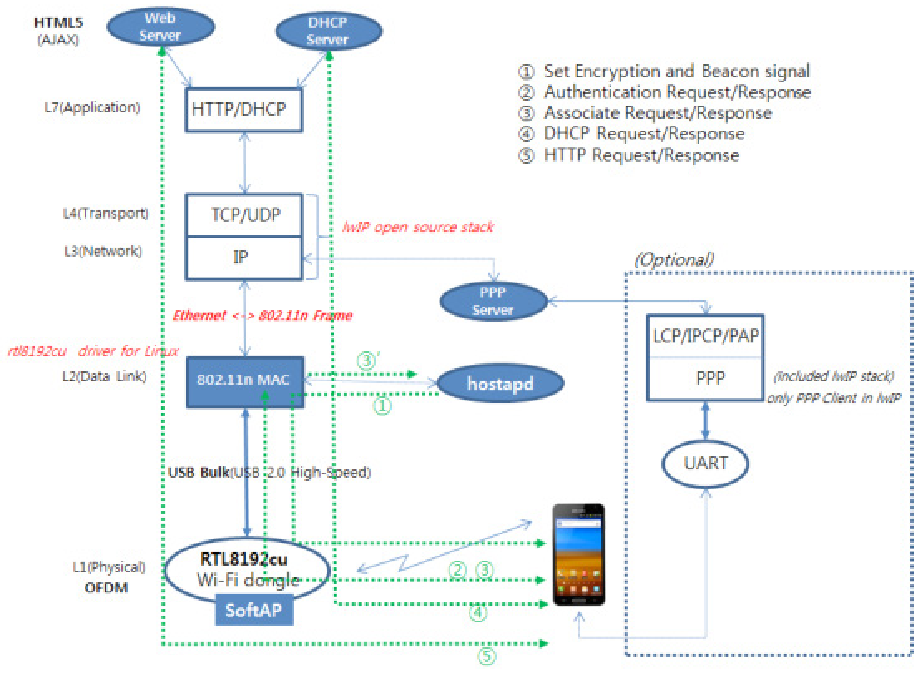


Figure 3. Overall system protocol schematic.

Table 2. Botton half in LINUX

Bottom Half	status	Characteristic
BH	delete	Delete form kernel 2.5
Task Queue	delete	Delete form kernel 2.5
softirq	support kernel 2.3	Dynamically created or impossibe to destroy Occupied only by the interrupt handler After processing the interrupt or after running the check of ksoftirqd daemon processes. Runned by checking the delayed softirq directly from the network and the SCSI subsystem
Tasklet	support kernel 2.3	Can be implemented through the softirq, but dynamically created or destroyed It provides a simple interface with a less stringent rule lock After registration via tasklet init (). If you run through the tasklet_, schedule () exits through the tasklet kill ()
Work Queue	support kernel 2.5	Always runs in the process context because the delayed work is passed to kernel threads Because the scheduling can be run in the process context, scheduling is possible and human sleep is also possible

- Set Encryption and Beacon signal and.
- Authentication Request and Response.
- Associate Request and Response.
- DHCP Request and Response.
- HTTP Request and Response.

2.4.2 Hardware Specifications

The followings are the characteristics of the AP chip used in this paper:

- CPU : Cortex-A8 1 GHz 1 Core (VFPv3 and NEON SIMD).
- GPU : Mali400.
- VPU : CedaX (H.264 1080p@60 fps encoding).
- RAM : DDR2/DDR3 up to 2 G.
- Flash : NAND up to 64 GB and NOR (using SPI).
- LCD/LVDS/HDMI Interface.
- SPI/I2C/USB2.0/SD Card Interface.

Figure 4 shows the developed hardware.

2.5 Development Results

As shown in the configuration diagram of the interface with the web server, the image input from the front and rear cameras is encoded in the H.264 video encoder and AAC audio encoder. Then it is stored in the SD card through the muxer as mp4 file format.

A smart phone is first connected to a Wi-Fi development board via Wi-Fi USB dongle to operate in a wireless AP (Access Point) mode. If you want to request an HTTP streaming video from a smart phone to connect

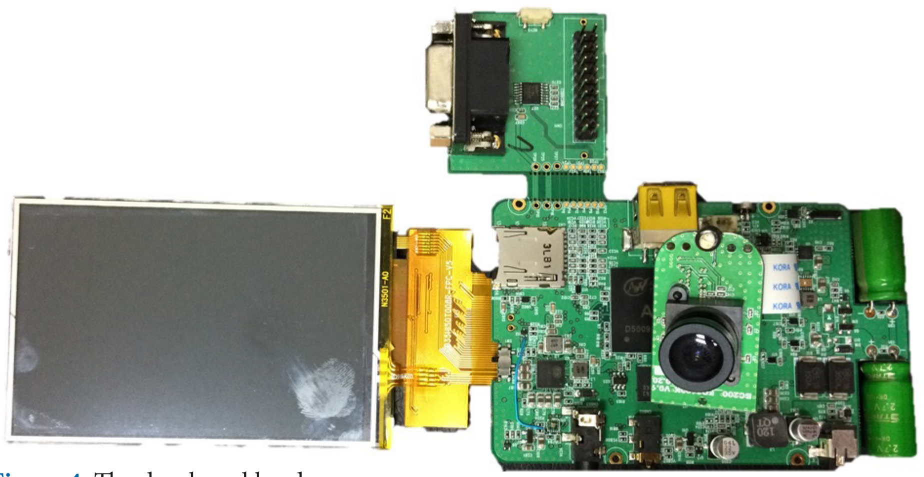
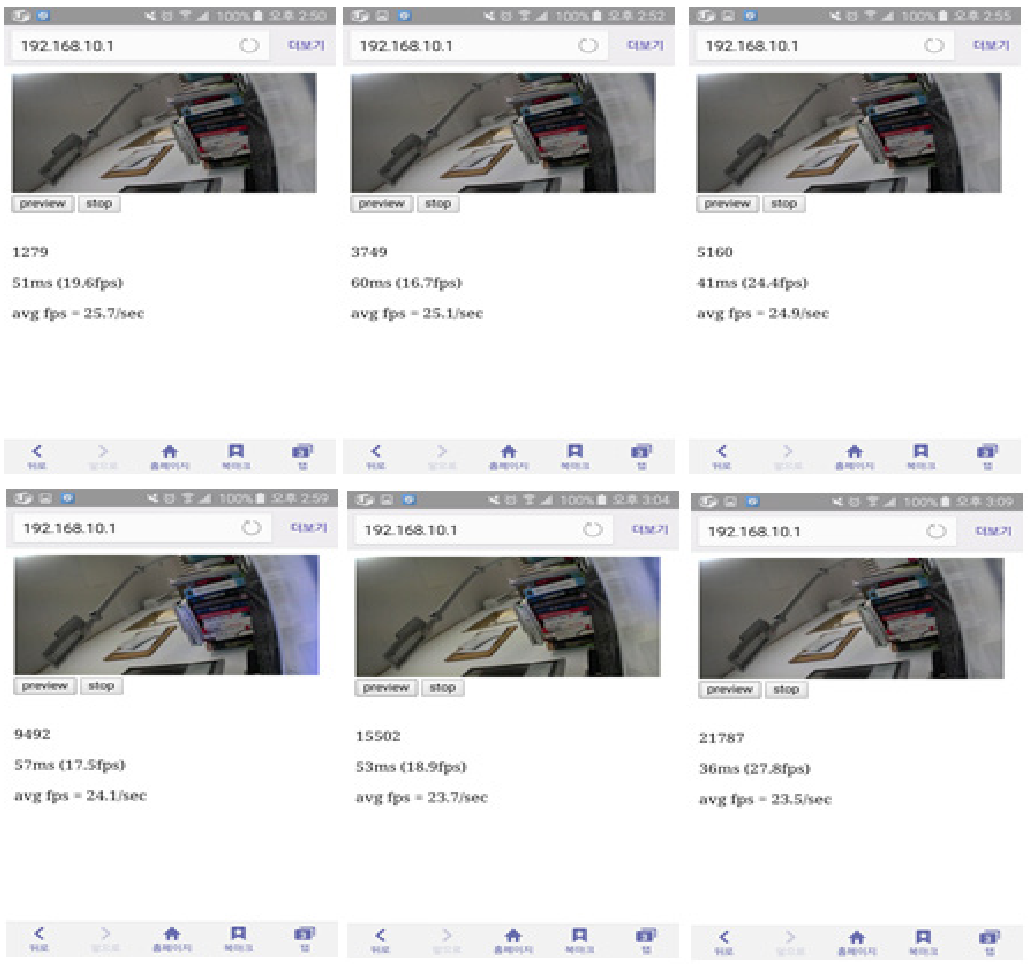


Figure 4. The developed hardware.



```
1279
51ms (19.6fps)
avg fps = 25.7/sec

3749
60ms (16.7fps)
avg fps = 25.1/sec

5160
41ms (24.4fps)
avg fps = 24.9/sec

9492
57ms (17.5fps)
avg fps = 24.1/sec

15502
53ms (18.9fps)
avg fps = 23.7/sec

21787
36ms (27.8fps)
avg fps = 23.5/sec
```

Figure 5. Smartphone measurement data.

to http://192.168.10.1 browser, using the libjpeg-turbo library YUV image output with the H.264 video encoder, convert to a JPEG image and send to the browser.

Images stored on the SD card are stored in X_YYYYMMDD_hhmmss_N file name and type. The first letter 'F' means front image, while 'R' means rear image. Recording time per file is typically 1 minute.

A browser screen captured at the front-to-back two-channel recording and measured with the basic browser of the simultaneous smart phones through the Wi-Fi at average number of video frames per second, is shown in Table 3 and Figure 5. The key technology as the results of this paper is as such, faster boot time, parts cost reduction and consumption of low dose of SPI NOR flash compared to Linux.

Table 3. Measurement data

Measurement time(min)	Average number of incoming frames per second (fps)
1	25.7
3	25.1
5	24.9
10	24.1
15	23.7
20	23.5

3 Conclusion

- Using RTOS technology through the development of this paper, we developed an MP4 encoder/decoder/muxer. Up to ten seconds of booting speed, minimum memory and the advantage of operating at lower power consumption than a general-purpose Linux by utilizing Wi-Fi dongle provide smartphone streaming functionality and lower product price per unit.

- WiFi with its widely known weakness of the RTOS system has been overcome by porting TCP/IP, Web server and DHCP server. Wi-Fi can be used even on the RTOS by implementing the protocol stack and combining them with the USB OTG Host Interface.
- There has been no other alternative than the use of NAND flash. This is due to Linux kernel size itself and with the program mounted on. As a result, the whole code size is increased and accordingly an increase in the cost per unit has been the point at issue. The key technology as the results of this paper is as such, faster boot time, parts cost reduction and consumption of low dose of SPI NOR flash compared to Linux. This can be implemented by using the RTOS-based platform that supports WiFi and lower processing power at the same time.

4. References

1. Karcanaj H, Bumci E, Tafa I, Fejzaj J. Deadlocks in Different Operating Systems. 2015 12th International Conference Information Technology - New Generations (ITNG); 2015 Apr. p. 717–8.
2. Bera R, Paul B, Guchhait A, Sil S, Sinha NB. Dogra S. Wireless embedded system for multimedia campus network utilizing IEEE 802.11 N (draft) and WiMax Radio. IFIP International Conference on Wireless and Optical Communications Networks (2007. WOCN `07; Singapore. 2007 Jul 2-4. p. 1–5.
3. Davis K. Linux Network Programming. Copyright ... Distributed to the book trade in the United States by Springer-Verlag: New York; 2004.
4. Choi J, Chae K, Jung S. Video data collection scheme from vehicle black box using time and location information for public safety. Journal of the Korea Institute of Information Security and Cryptography. 2012; 22(4):771–83.
5. IRS Grobal. 2012. Available from: http://www.irsglobal.com
6. Axelson J. USB complete: The developer's guide. 3rd ed. USA: Lakeview Research; 2006. p. 535–49.
7. Yetgin Z, Seckin G. Progressive download for multimedia broadcast multicast service. IEEE Multimedia. 2009 Apr–Jun; 16(2):76–85.
8. Dinu, DD, Togan M. DHCP server authentication using digital certificates. 2014 10th International Communications (COMM) Conference; Bucharest. 2014 May 29-31. p. 1–6.
9. Jangju K, Jongwook J. Sung–Hyun B, You-Sin P. An implement of smart Black box with GPS and Wireless network. KISTI; 2010. p. 217–26.
10. Kim J, Lee I. WLAN: History and new enabling MIMO techniques for next generation standards. IEEE Communications Magazine. 2015 Mar; 53(3):134–40.
11. Kim D. Wi-Fi On. Korea: Midasbooks; 2011.
12. Labrosse JJ. MicroC/OS-II. The Real-Time Kernel. 2nd ed. USA: CMP Books; 2002.
13. Murikipudi A, Prakash V, Vigneswaran T. Performance analysis of real time operating system with general purpose operating system for mobile robotic system. Indian Journal of Science and Technology. 2015 Aug; 8(19):1–6.
14. Chakravarthi MK, Vinay PK, Venkatesan N. Design and simulation of internal model controller for a real time nonlinear process. Indian Journal of Science and Technology. 2015 Aug; 8(19):1–6.