Multimedia Systems (2011) 17:299–311
DOI 10.1007/s00530-010-0211-z

REGULAR PAPER

Design and integration of the OpenCore-based mobile TV framework for DVB-H/T wireless network

Chin-Feng Lai · Yueh-Min Huang · Jiann-Liang Chen · Wen Ji · Min Chen

Published online: 25 November 2010

Abstract In the array of mobile communication techniques, the application of a mobile phone combined with television is a new technique under development. As TV program is a real-time video/audio service, in comparison with either traditional video/audio file downloads or network video/audio streams, there are more technical difficulties to be overcome, in particular, how to satisfy the playback functions of TV programs in hand-held device. OpenCore is a multimedia framework, which has recently been widely applied in hand-held devices, but it does not offer functions of mobile TV. To solve this problem, this study incorporates the function of mobile TV into the OpenCore framework, in order to support both formats of TV signals, i.e. DVB-H and DVB-T. The incorporated function, DVB-H/T, has different characteristics, so that users can select TV signals according to their receiving environments and fulfill their needs in TV programs selection.

Keywords OpenCore · Mobile TV · DVB wireless network

C.-F. Lai
Institute of Computer Science and Information Engineering, National Ilan University, Ilan, Taiwan
e-mail: cinfon@ieee.org

Y.-M. Huang
Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan

Y.-M. Huang (✉)
Department of Applied Geoinformatics, Chia Nan University of Pharmacy and Science, Tainan, Taiwan
e-mail: huang@mail.ncku.edu.tw

J.-L. Chen
Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan
e-mail: lchen@mail.ntust.edu.tw

W. Ji
Department of Computing, Hong Kong Polytechnic University, Hong Kong, Hong Kong
e-mail: jiwen@ieee.org

M. Chen
School of Computer Science and Engineering, National Seoul University, Seoul, Korea
e-mail: minchen@ieee.org

1 Introduction

With the advancement of digital TV and transmission technology, watching TV without the boundary of time and space by using various wireless terminal devices has become an inevitable trend. At the early stage, digital video broadcasting-terrestrial (DVB-T) was designed for receivers without mobile devices, and power consumption was not a major problem; thus, it is unsuitable for mobile device. Digital video broadcasting-handheld (DVB-H) is a digital video signal transmission technique compatible with the DVB-T mode and suitable for hand-held receivers as shown in Fig. 1 [1–4]. Up to the present, however, as DVB-T broadcasting system is completely distributed, broadcasting contents do not require extra charge, and with increasingly reduced power consumption in IC at RF front end, DVB-T can also be used in mobile phones to receive TV programs [5].

For applications of mobile devices, developers must consider limited memory space and computing power when designing applications on specific platforms. Thus, it

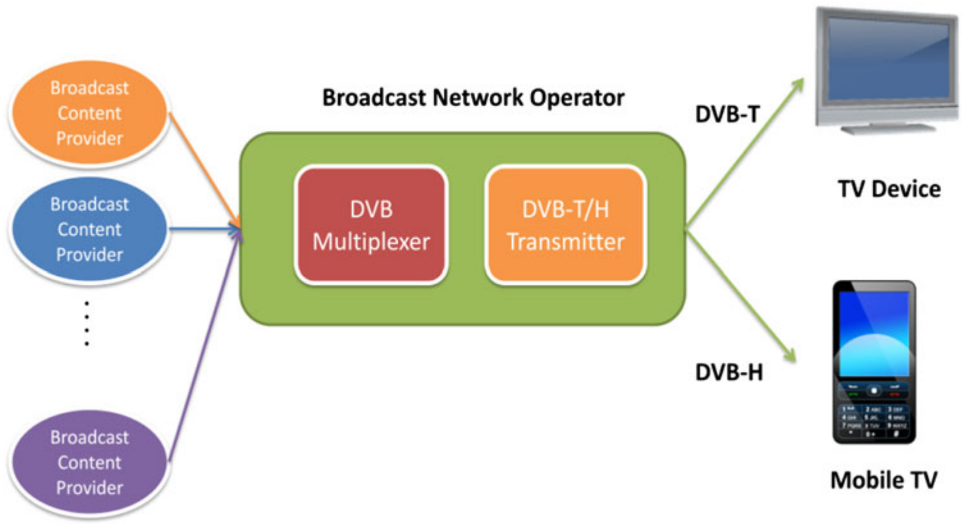


Fig. 1 DVB-H/T broadcast network operator

mostly makes development progress spend much time and money. For this reason, perfective frameworks that provided users the defined API being irrelevant to platform features are required to integrate hardware and software co-design. OpenCore is a multimedia framework based on PacketVideo's CORE and can provide the necessary media features for the development of media equipment; hence, it has been widely applied in the multimedia players of mobile equipment. However, it does not support the function of TV program player.

Thus, this paper proposes an OpenCore-based mobile TV framework, which incorporates DVB-H and DVB-T into OpenCore. DVB-H/T mobile TV realization consists of two main parts: OpenCore DVB-H/T parser node and OpenMax DVB-H/T component. The former is used to interpret relevant information in DVB-H/T signal, extract, and then send audio/video data to the decode node, while the latter enables communication between OpenCore and the OpenMax DVB-H/T component.

The remainder of this paper is organized as follows: Sect. 2 discusses relevant technologies of OpenCore and DVB-H/T; Sect. 3 describes the complete OpenCore-based mobile TV framework, including flow and methods of OpenCore DVB-H/T parser node and OpenMax DVB-H/T component; Sect. 4 presents practices and results; Sect. 5 gives conclusions.

2 Related work

2.1 Mobile TV

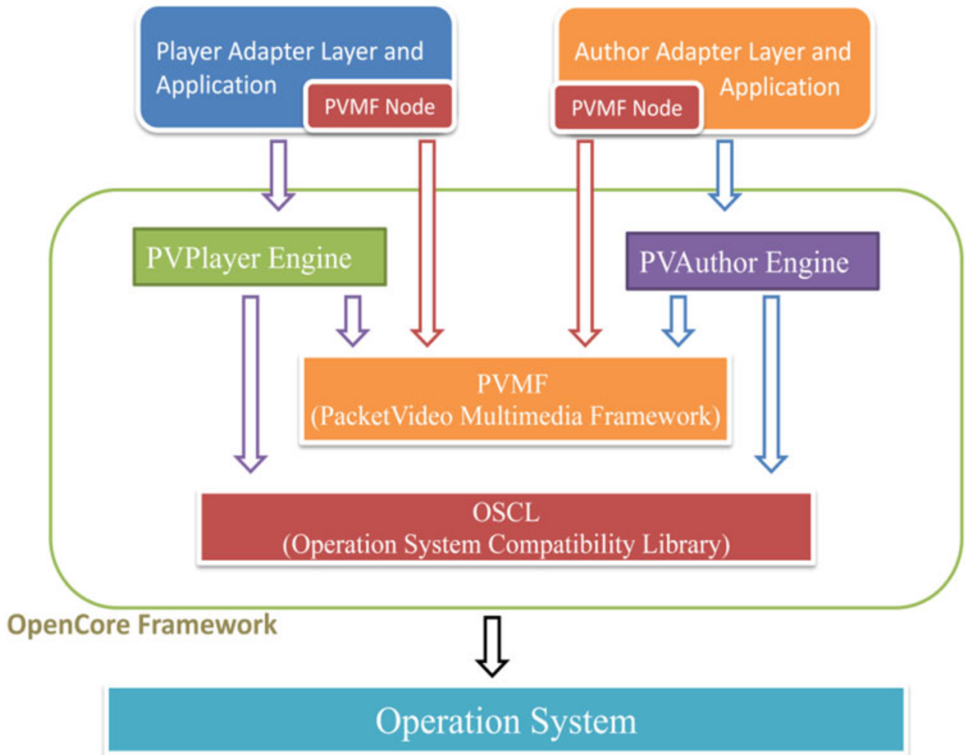
Mobile TV usually denotes that TV programs can be watched on hand-held devices and has been developed as "mobile-centric". At present, two methods are available to watch TV programs on hand-held devices: (1) the manufacturer can transmit TV programs to the mobile phones users in the form of 3G stream [6–8] and (2) digital video broadcast can be realized through terrestrial TV station transmission systems [9]. The main difference between the two methods is that 3G transmits signals through wireless communication and, thus, may be subject to limitations in transmission rates and band width. Digital video broadcast transmits TV programs through terrestrial TV station transmission systems; thus, the problem of transmission network congestion may be avoided. As the broadcast networks starts making the content available online, there are many standards for digital video broadcasting. Digital Video Broadcasting (DVB) is utilized in Europe and Asia, while Advanced Television Systems Committee (ATSC) is used in North America.

About the researches on mobile TV, most papers either focus on the transmission control between front-end transmitters and receivers for mobile devices to maintain video quality and signal reliability or propose interactive service mechanisms with TV return channels [10–14]. However, the challenges on mobile devices are discussed less frequently, especially when mobile TV services become more and more popular. Due to the different characteristic of each platform, developers spend much time in attempting to overcome hardware limits. Therefore, this study integrates mobile TV services that support DVB-H/T format into the OpenCore framework to reduce development time.

2.2 OpenCore

Figure 2 shows OpenCore, also known as PacketVideo. In fact, PacketVideo is the name of a company, and OpenCore is the name of the software layer of the multimedia

Fig. 2 OpenCore architecture



framework. In Android development, both have the same meaning. In comparison with other link libraries of Android, OpenCore has a large number of codes, which are realized based on C++ and defines all-functional porting layers of the operating system. Various basic functions are encapsulated as categories, and most interfaces among the different layers apply modes, such as succession. The OpenCore framework can be separated into the following components in terms of its functions.

PVPlayer: It provides functions of a media player and complete playback functions of various audio/video streams.

PVAuthor: It provides functions of media stream recording and complete functions of capturing audio/video streams and static images.

PVPlayer and PVAuthor: They are provided to the developer in the form of SDK, on which several application programs are based, and services can be established. Common multimedia application programs applied in the mobile terminal include a media player, a camera, a video recorder, and a sound recorder.

To better organize the whole framework, in the macro level, software layers of OpenCore are separated into several layers:

The Operating System Compatibility Library (OSCL) includes some basic operations of the operating system in order to better realize porting among the different operating systems. These include base data type, configuration, character string tool, IO, error treatment, execution, etc.

PacketVideo Multimedia Framework (PVMF): It consists of a parser, a composer, and an encoding/decoding node; it may replace its universal interface, thus realizing nodes in the user layer.

PVPlayer/PVAuthor Engine: It provides interfaces necessary for transmitting data packets between application programs and the OpenCore framework, and combines with the node to realize such functions as encoding/decoding and stream control.

In fact, OpenCore has many contents; from the viewpoint of playback, PVPlayer input (Source) is file or network media stream, and output (Sink) is a video/audio output device, its basic functions including media stream control, file parsing, and video/audio stream decoding. In addition to playing media files, it also includes an RTSP stream connected with a network (Real Time Stream Protocol). In terms of media stream recording, the PVAuthor input (Source) is a camera, a microphone, and other devices, and output (Sink) is various files, including streaming synchronism, video/audio stream encoding, and file writing.

When OpenCore SDK is used, it may be necessary to provide an Adaptor in the application program layer, based on which specific functions are realized. The PVMF node can also rely on universal interfaces, realized in the upper layer, and used in the form of a cheat program.

2.3 DVB-H/T

The DVB-H is based on DVB-T and incorporates solutions that can reduce the power consumption of the receiving set and improve mobile receiving performance [15, 16]. Figure 3 presents an overview of the DVB-H/T system specifications. In general, TV programs broadcasted in the DVB-T signal transmission mode are converted by the video/audio A/D converter transmission terminal, from analog video and audio signals into digital signals, which then transforms TV programs into MPEG-2 TV service through video coding and audio coding by applying MPEG-2 encoding/decoding techniques [17, 18]. DVB-H service data are compressed and encapsulated into IP packages, and then further encapsulated in a transport stream through MPE mechanisms. After time slicing data stream is added, an IP package is multitasked together with other DVB-T TV services through a multiplexer into a large multiple program transport stream. Then the data are sent out after being changed into a wireless signal. At the receiving end, if users intend to receive services, the front circuit of the receiver should be in all-time operation in order to decode the complete transport stream. Subsequently, the de-multiplexer is used to extract video signals, audio signals, and data streams of the programs selected by the user, which are then sent to the video decoder, audio decoder, and other application programs, respectively, for processing. MPE-FEC and time slicing at transmission terminals are collectively referred to as DVB-H IP-Encapsulator, and the reverse decoding part of the receiving terminal is called a DVB-H IP-Decapsulator [19, 20].

The encapsulating format of the whole DVB-H/T and the encapsulating format of IP data of various layers of the whole DVB-H are shown in Fig. 4. IP packages are inside the MPE Section, and redundant data are inside the FEC Section. After the Section formats are encapsulated, they are end-to-end connected in the encapsulating format, within the MPE Section and FEC Section into Section data stream. Following this, each Section data stream is cut into 184-byte intervals from its first byte, and a 4-byte transport stream file head is added to each 184 bytes data length at its head, in order to complete one complete transport stream package. In other words, the length of MPEG2 Transport Stream Packet is of 188 bytes, and it consists of two parts, a data Header, occupying 4 bytes, and providing such information as Sync. Information such as Byte = 47hex, error indications, and stream package identifications have a length of 184 byte in the second part of the transfer data payload. The IP and UDP packages contain package file headers. The RTP package is encapsulated according to RFC3984, and carries H.264 images and AAC compressed voices. The DVB-T has relatively simple package format. After the MPE Section is decoded, the MPEG-2 images and AC-3/MPEG voices will occur.

3 Proposed architecture

3.1 Overall architecture

At present, audio/video types supported by OpenCore include MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF, etc. Therefore, it is necessary to add the DVB-H/T file format, DVB-H/T parsing components to the OpenCore framework. In the OpenCore framework, the MEPG-2 data structure is defined, and developed DVB-H/T components are added to the original OpenCore framework as shown in Fig. 5, and its development flow is described as follows:

1. Adjust modem parameters, such as frequency, quadrature amplitude modulation (QAM), bandwidth, transmission rate, and guard interval. Transport streams can be smoothly received only through modulating used parameters, and then acquired stream packet information is transmitted, until the Source Node is completed.
2. Parse transport streams:
   (i) Reject incomplete packages.
   (ii) Apply DVB-H/T file format parser, as completed in PVMF, to parse videos, audios, captions, and multilingual stream PID from the data stream transmitted by the Source Node.
   (iii) Processing mode of video stream: As the unit stream is in the DVB-H/T compressed format, decoding is required for playback. Therefore, the DVB-H/T decoding algorithm is used for decoding video unit streams.
   (iv) Processing mode of audio stream: Same as the video stream, the MPEG2 Layer1 decoding algorithm is used for audio unit stream servers for decoding.
3. Decoded and adjusted frames are sent to the constructed Sink Node. Here, the video frame buffer, which is housed within the system memory section, corresponds to the memory of solid display. The video decoder converts the digital data inside the frame buffer into analog composite signals and output to the screen.
4. After decoding, audio waveform data are sent to the constructed Sink Node through a sequence audio port, which is an audio output port.
5. The A/V synchronous mechanism established by the OpenCore framework is used to simultaneously send out images and voices.

In digital television, all contents such as video, audio, text, and image are digitalized, and then the digital data are

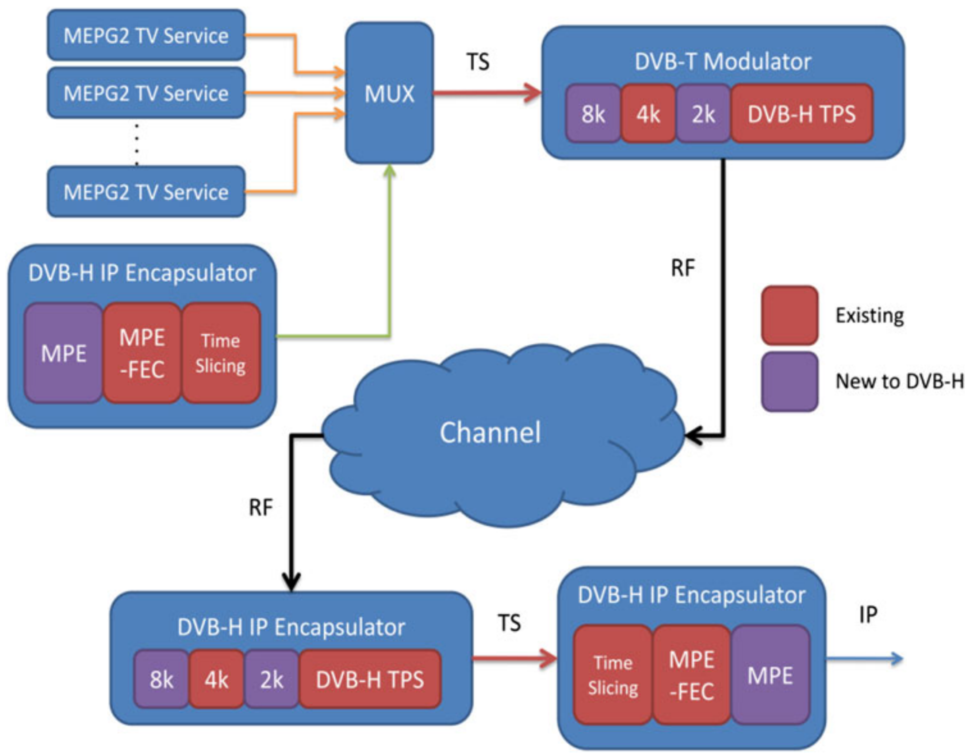


Fig. 3 DVB-H/T system specifications

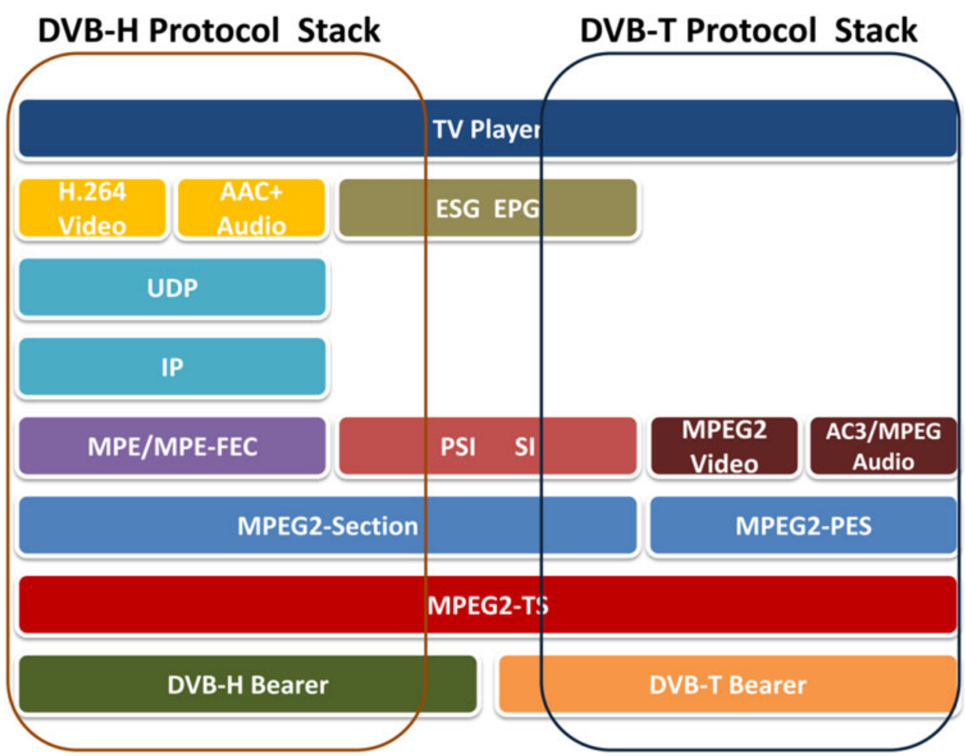


Fig. 4 DVB-H/T encapsulating format

compressed and packed according to the DVB standard. This generates fixed length (188 bits) data packages to transfer. The data packages are reused to generate transferring stream (TS). In a general case, one channel can only map to a TS. A TS stream of a channel is composed of several programs and transactions. The stream format of the whole DVB-H/T signal is shown in Fig. 6. In this figure, we can realize the conversion of DVB-H/T format clearly. As OpenCore does not support DVB-H/T TV signals, DVB-H/T signal characteristics are added to the

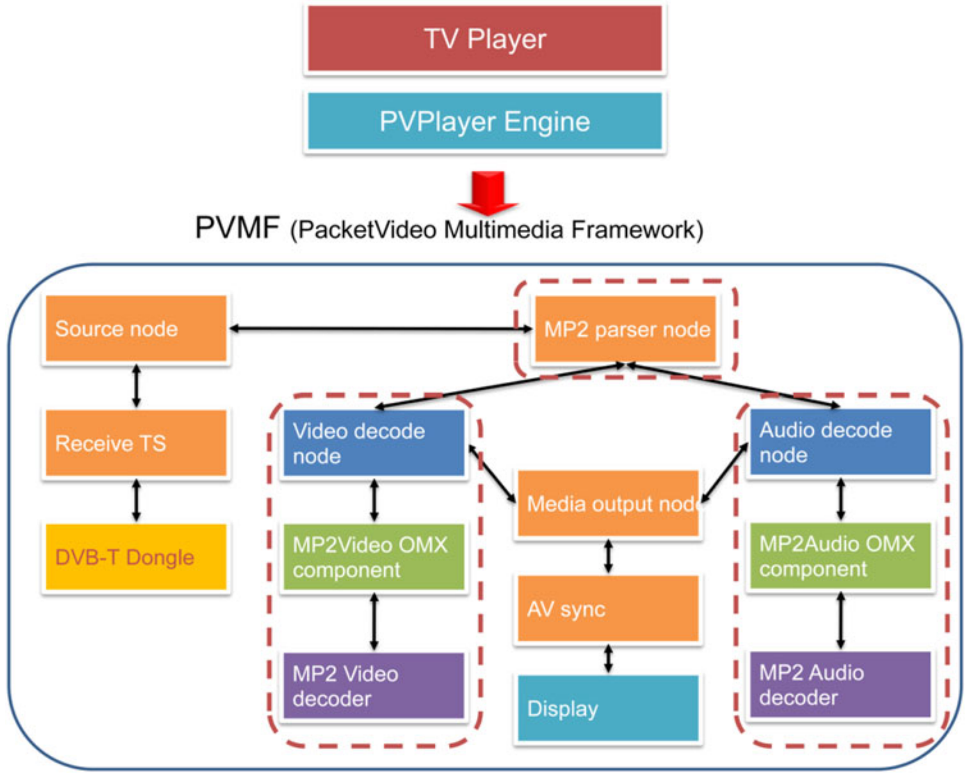


Fig. 5 DVB-H/T component of OpenCore

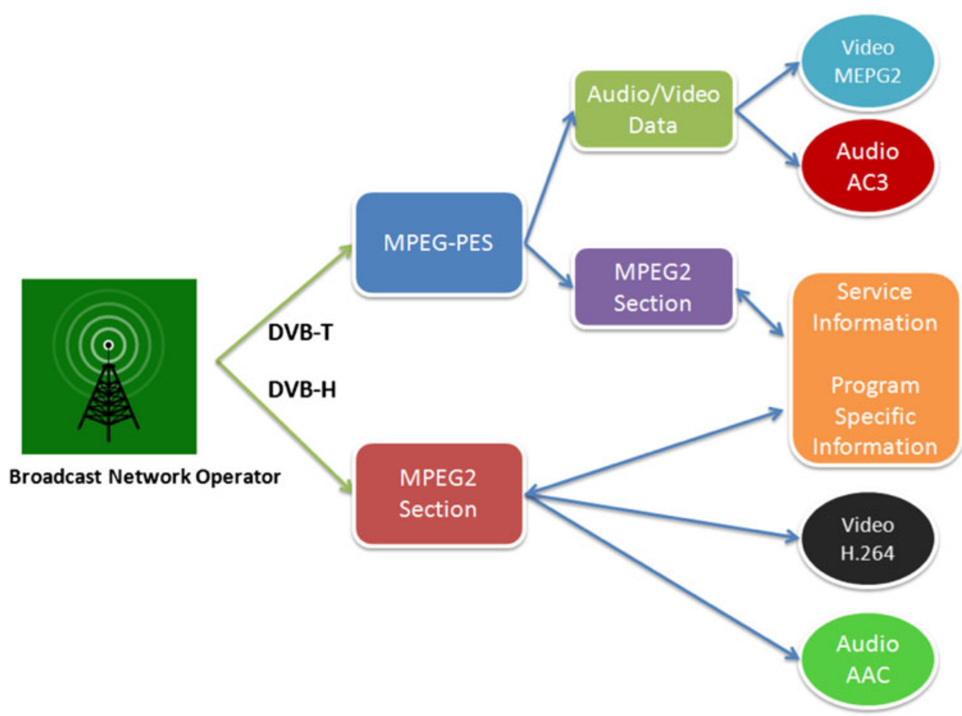


Fig. 6 DVB-H/T signal data format

OpenCore framework, in which most important parts requiring modification include two components, the parse node and OpenMax components. They are described in detail as follows.

3.2 DVB-H/T parser node

The Parser Node is used to parse DVB-H/T TV signals continuously transmitted by the Source Node into TV programs, PEG, and accompanying information in the aforesaid two specifications.

The DVB-T signal is sent to a mobile TV device in the packet format of MPEG2-PES (Packetized Elementary Stream), and its composition is shown in Fig. 7. Each package length is fixed at 188 bytes, including the synchronization bit, state flag bit, packet identifier (PID), dispensable modulation field, and transport field, and the others are payload. The length of the transport stream packet is fixed at 188 bytes. The transport stream is data stream produced by a multitasked unit stream consisting of several programs and data. One program may comprise one video unit stream, one or two audio unit streams, dispensable captions, or multilingual unit streams. A program Map Table (PMT), describing program information, is also included in the multitasked transport streams and has a unique PID. The receiving terminal de-multitasks program corresponds to the table of transport streams and obtains the program PID from the corresponding table in order to extract specific program streams and corresponding service information (SI) and program-specific information (PSI). In DVB-H, the streams are transmitted in the format of MPEG2-Section. Relevant information in the Section Header, such as table_id, Network_ID, length, section_no, and other flags, are used to analyze streams, SI, and PSI. Four tables are included in PSI: the program association table (PAT), the program map table (PMT), and the conditional access table (CAT), and network information table (NIT). MPEG specifications do not define the details CAT or NIT, which are defined by the manufacturers. PAT records the list of all programs in the streams and gives each program one 16-bit program code. Each program has one PID of program corresponding table, and the PID of the program index table is defined as 0. See Fig. 8 for examples. From the packet with a PID of 0, it may be found that several programs are included in the transport streams. PID codes of corresponding tables of each program, information of PMT recording programs, video unit streams PID of programs, audio unit stream PID, and caption unit stream PID are included in this table. Figure 9 shows an example. The relationship between PAT and PMT is shown in Fig. 10. After the data are parsed by the Parser Node, Audio/Video Data are sent to Audio/Video Decode Node, and EPG is shown on the TV Player Interface.

3.3 OpenMax DVB-H/T component

OpenMax can enable a platform with hardware encoding/decoding functions to provide a uniform interface and

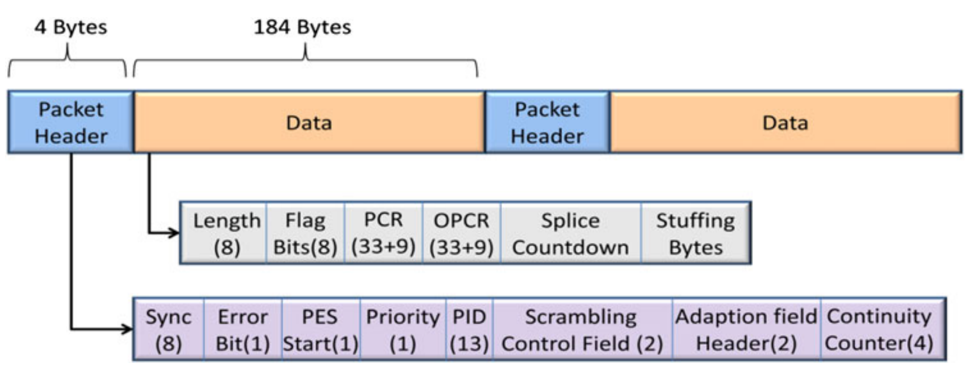


Fig. 7 MPEG2-PES format

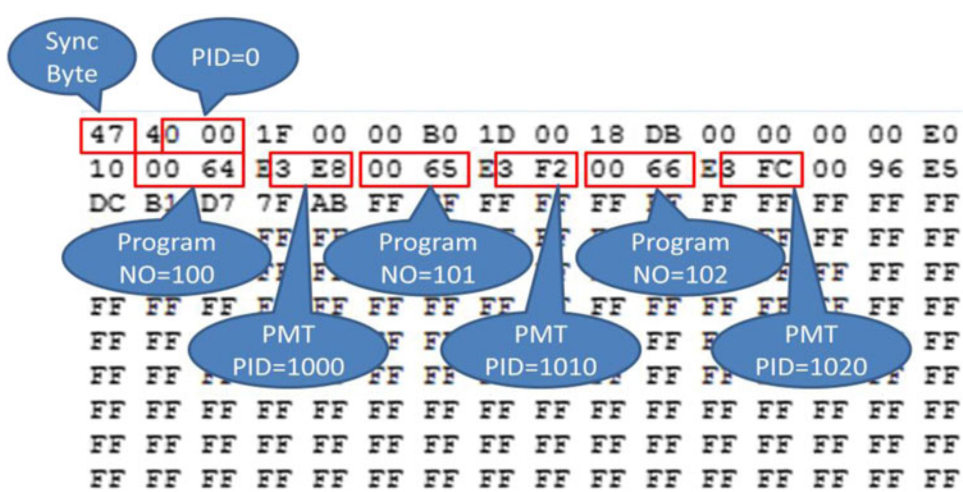


Fig. 8 PAT data example

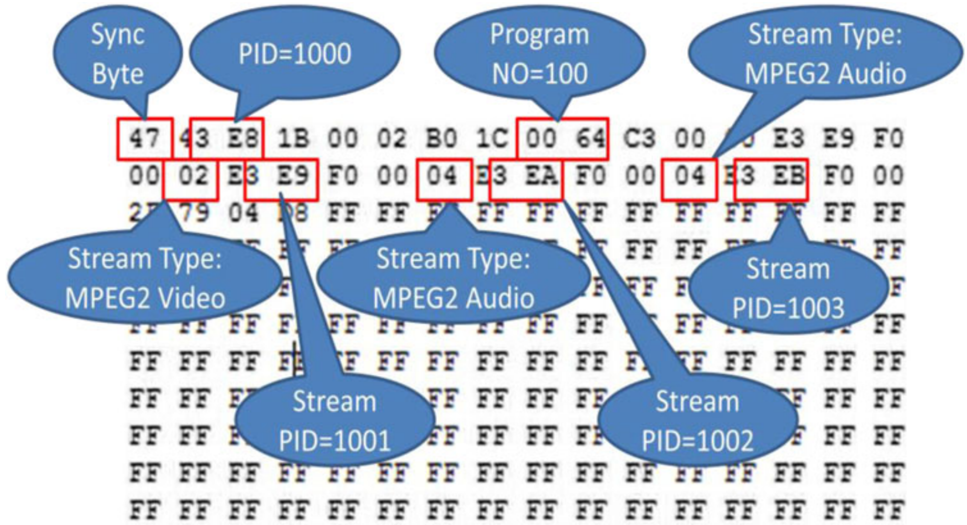


Fig. 9 PMT data example

PAT PID=0

Program number	PMT PID
0	2000
1	5000
.	.
.	.
100	1000

PMT PID=1000

Stream type	PID
MPEG2 video	1001
MPEG2 audio	1002
MPEG2 audio	1003
.	.
Teletext	100

Fig. 10 Relationships between PAT and PMT

framework, which uses hardware acceleration for encoding/decoding, and maintains a uniform interface with the outside. In the OpenCore framework, API of OpenMax IL (Integration Layer) is included to provide multimedia encoder/decoder functions in PVMF video decode node. OpenMax IL technical specifications define the media component interface, which quickly integrates the hardware transcoder in the stream media framework of the embedded equipment. As OpenCore has no actual OpenMax component of DVB-H/T, official specifications for OpenMax IL should be referred to during development, in order to produce DVB-H/T components and API. In this way, OpenMax IL may be used in such way that the DVB-H/T encoder/decoder will provide a system abstract interface for the convenient use in calls of the base DVB-H/T encoder/decoder of the developer as shown in Fig. 11.

The call sequence of OpenCore and OpenMax components shown in Fig. 12 may be roughly described by the following drawings.

All these processes are controlled by PVMF, which gives commands to the control state conversion of the OpenCore component. Interaction processes between OpenCore and the OpenMax core/component are divided

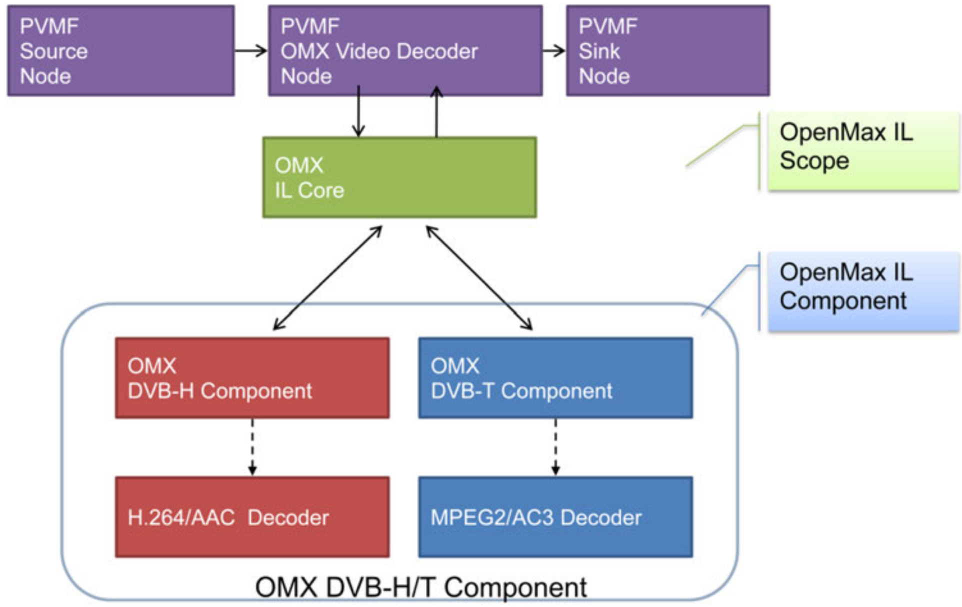


Fig. 11 OpenMax DVB-H/T component

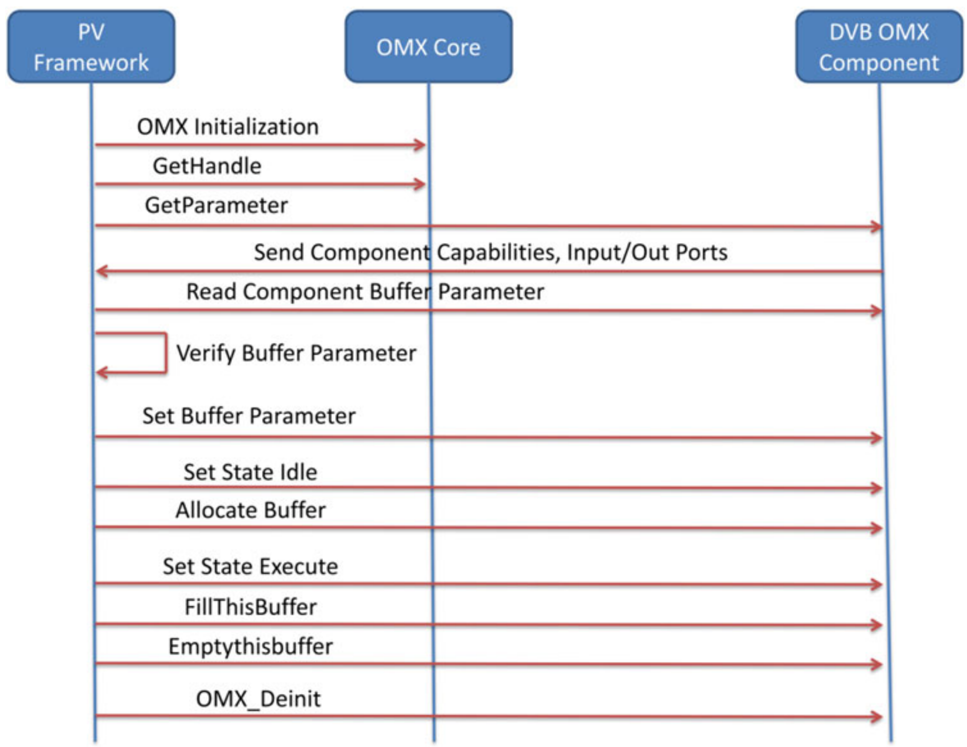


Fig. 12 Call sequence of OpenCore and OMX components

into the following processes, all of which are realized based on the typical API sequence.

1. Calls of OMX, by PVMF, commence from the initialization of the OMX core. PVMF calls OMX_Init to initialize the OMX core, requests all the possessed components from OMX core, and then the valid component roles based on the acquired component name, and last, establishes one registry of valid components and their roles.
2. After initializing the OMX core, PVMF should, based on the acquired component name, initialize the OMX component, list the capabilities of the corresponding component, and obtain a valid port number of the Audio/Video component. According to such

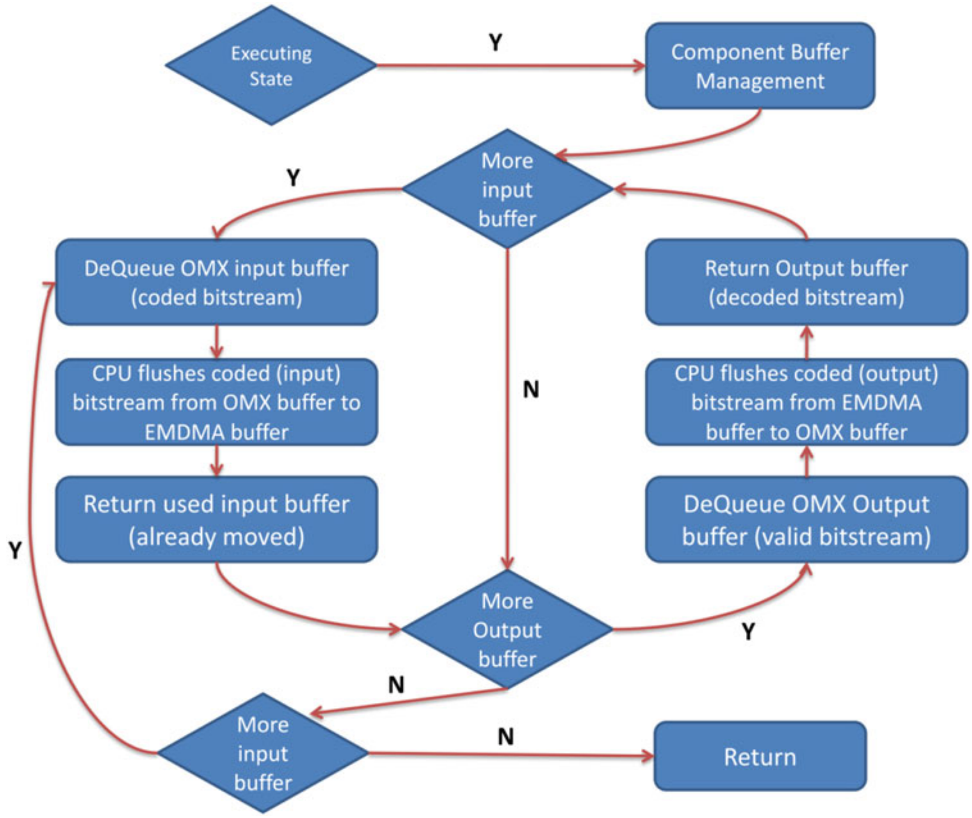


Fig. 13 DVB decode component flow

capabilities, whether the component supports part of the frame and whether the component supports Use-buffer or AllocateBuffer will be known.

3. The PVMF communicates with the OMX component input/output buffer after initializing both OMX core and OMX component. OMX_GetParameter is first used to obtain input/output port buffer parameters, then the obtained parameters are verified, and last, the OMX_SetParameter is called to set buffer parameters for the input/output port.
4. After the initialization and communication with input buffer, the OMX component is loaded, and then enters into an idle state. Then PVMF will allocate buffers for all input/output ports.
5. After entering into an idle state, The OMX component will move into an execution state to process data and conduct a data exchange with the PVMF. First, PVMF gives commands to the OMX component to transform the state from Idle to Execute. Next, the PCMF will send data to a certain input port of the component, through OMX_EmptyThisBuffer, forcing the component to acquire data from the buffer for processing. At the same time, PCMF requires the component to fill decoded or encoded data into its one output port, through FillThisBuffer. Before PCMF uses emptythis-buffer to re-allocate the buffer to the component, the component cannot operate the buffer.
6. At last, PVMF calls OMX_Deinit() to release the OMX core.

The DVB-H/T data are transmitted to a DVB Decode Component for decoding Audio and Video. First, the DVB Decode Component will inquire through OpenMax IL Core, upon intervals, whether the OMX node has data for decoding. If it has, OpenMax IL Core will clear the required OMX Input buffer and store the data in the OMX node for decoding in the OMX buffer. Then data in OMX Input buffer are migrated to EMDMA Buffer for decoding, and then the OpenMax IL Core will continually inquire whether EMDMA Buffer decoding is finished. After the decoding, the OpenMax IL Core will immediately clear the required OMX Output buffer, and then transmit decoded data in the EMDMA Buffer to the OMX Output Buffer, and then inquire whether there are data in the Input Buffer for decoding. The Component will continually repeat this loop until no data are available for decoding. Figure 13 is the process of whole DVB Decode Component.

4 Prototype implementation

This section will present the OpenCore-based Mobile TV framework on Android Mobile Operator System and

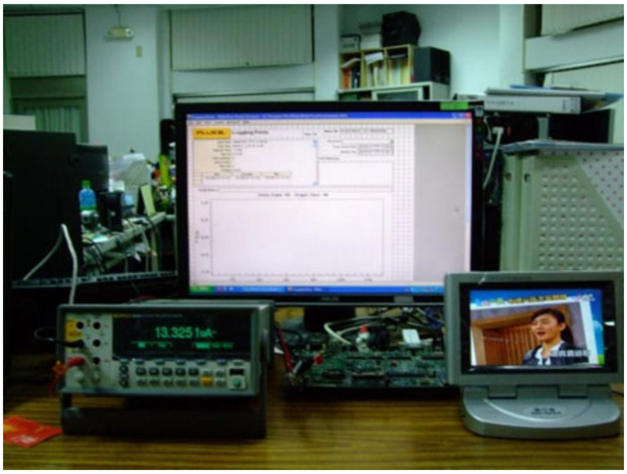


Fig. 14 Mobile phone simulation platform

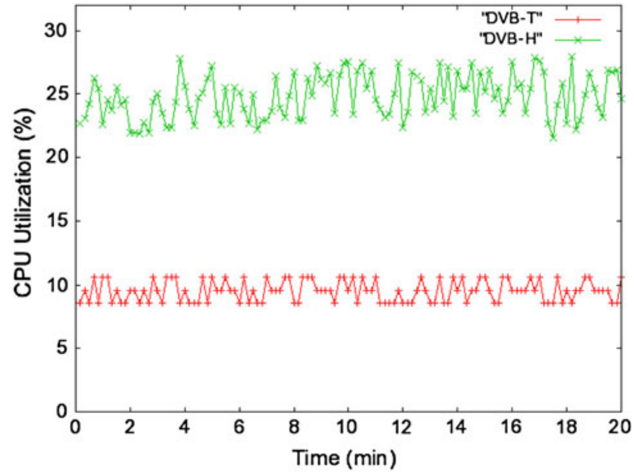


Fig. 15 CPU utilization of OpenMax DVB-H/T component

apply a 2.4 GHz four-core processor computer as a mobile phone simulation platform as shown in Fig. 14. Video resolution of the tested DVB-T and DVB-H signals is regulated at 352 × 288, and the frequency rate is 20 fps. Data quantity required for a single channel of DVB-H is about 350–490 kbps, and data quantity required for single channel of DVB-T is 720–960 kbps. An 8-MHz radio channel with 16-QAM modulator is adopted, and a guard interval is set at 1/8. Received DVB-H and DVB-D signals are parsed and played back, and the operational efficiency of the platform is measured. Then this paper measures the power consumption of these two factors in a mobile TV, including program playback.

4.1 Operation efficiency

As this platform receives TV signals from the demodulator card of DVB-H/T, in assessment of operational efficiency, this study excludes demodulator loading, and directly measures efficiency of the receiving signals from the source node to the parser node, and sending audio/video signals to the OpenMax DVB-H/T component for decoding, as shown in Fig. 15. This study then tests a single TV program channel for 20 min and measures the value at 10-s intervals. The results indicate that as DVB-T Video applies the MPEG2 compression method, DVB-H adopts the H.264 compression method, and the H.264 compression rate is 2–3 times the MPEG-2 compression rate; however, the decoding process operation quantity is also raised, and the reason for the fluctuation in the DVB-H curve is that the reference mode of H.264, in the respect of frames, is different from that of MPEG2.

4.2 Power consumption

To clearly measure the consumed power when this framework plays back TV programs, this study tests the power consumed when all TV programs are played back, and the mobile system is in an idle state, as shown in Fig. 16. When the system is out of service, consumed power is 0.9 W. Figure 17 shows the consumed power when DVB-H/T TV signals are played back. As seen, as DVB-H applies the Time Slicing technique, namely, only during the period when signals are sent (burst duration), the receiver is turned on to receive, and otherwise, it is in off state. This method greatly reduces the power consumption of the DVB-H Header (turner plus demodulator); however, operations, such as compression of H.264 at rear end, and filtering at MPE-EFC and IP Header would increase the

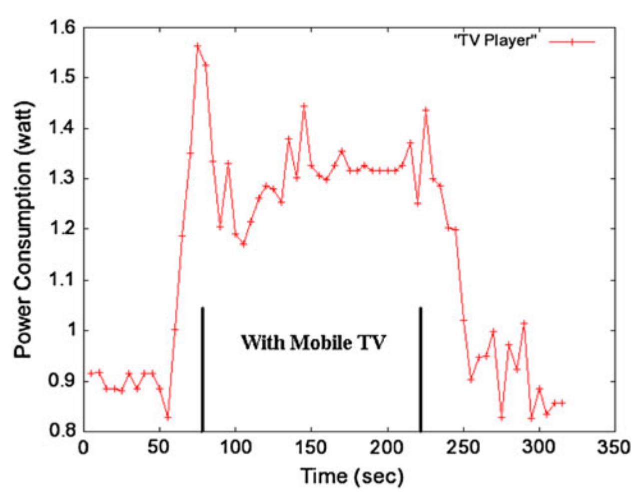


Fig. 16 Power consumption of the mobile TV

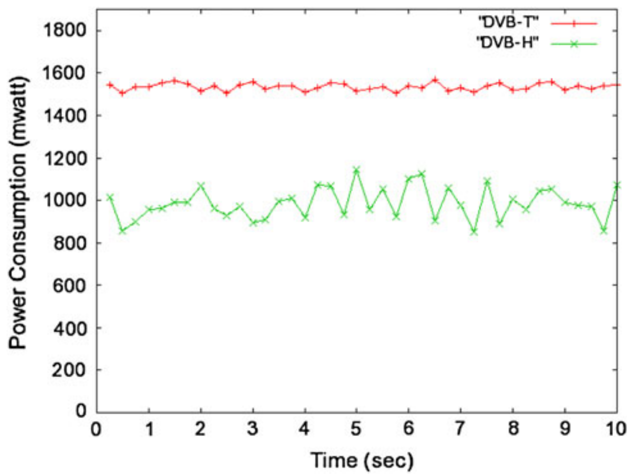


Fig. 17 Power consumption of OpenMax DVB-H/T component

DVB-H power consumption, but as a whole, power consumption of DVB-H is less than that of DVB-T.

4.3 Performance analysis

To compare with other mobile TV devices in performance, Video LAN Client (VLC), a famous multimedia player, was used to receive DVB data and play TV programs in Linux environment. In Fig. 18, the frequency rate of VLC is 19–20 fps, and OpenCore's is 14–15 fps. The results indicate that OpenCore, the defined framework architecture, must be followed to parse and decode DVB data. It makes the performance of OpenCore less than the contrast. However, the standard API of OpenCore framework could reduce the development time and cost greatly since the developers do not consider differences between platforms.

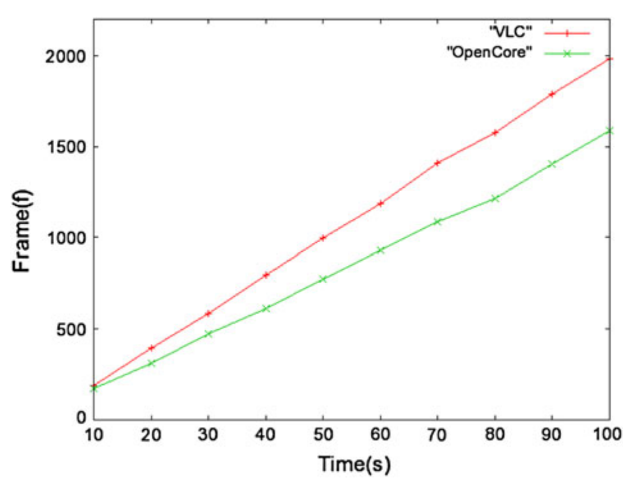


Fig. 18 Performance analysis of the mobile TV

5 Conclusion

This paper presents a mobile TV platform on the OpenCore framework, and design different processing flows for two TV signal forms, i.e. DVB-H and DVB-T, and explains how transport streams received by the hardware converter are converted and parsed into original data. In addition, according to OpenMax specification, it provides one Mobile TV component, communication interface between framework API and decoder, to decode into a playable format. The current DVB-T format has been improved in hardware technique, but not widely applied in the DVB-H basement station; hence, with this study, we can save the time for overcoming hardware limits and provide users with more feasible TV services in the future.

References

1. Zhou, L., Xiong, N., Shu, L., Vasilakos, A., Yeo, S.-S.: Context-aware multimedia service in heterogeneous networks. IEEE Intell. Syst. **25**(2), 40–47 (2010)
2. Cosmas, J., Itegaki, T., Cruickshank, L., et al.: Converged DVB-T and GPRS service scenarios and application production tools. In: Proceedings of CONFTELE 2003, Aveiro, Portugal (2003)
3. Zhou, L., Geller, B., Zheng, B., Wei, A., Cui, J.: Distributed resource allocation for multi-source multi-description multi-path video streaming over wireless networks. IEEE Trans. Broadcast. **55**(4), 731–741 (2009)
4. Zhou, L., Wang, X., Tu, W., Mutean, G., Geller, B.: Distributed scheduling scheme for video streaming over multi-channel multi-radio multi-hop wireless networks. IEEE J. Sel. Areas Commun. **28**(3), 409–419 (2010)
5. Angrisani, L., et al.: Power measurements in DVB-T systems: new proposal for enhancing reliability and repeatability. IEEE Trans. Instrum. Meas. **57**(10), 2108–2117 (2006)
6. Fuchs, H., Farber, N.: Optimizing channel change time in IPTV applications. In: Proceedings of IEEE Broadband Multimedia, Las Vegas (2008)
7. Parkvall, S., et al.: Evolving 3G mobile systems: broadband and broadcast services in WCDMA. IEEE Commun. Mag. **44**(2), 68–74 (2006)
8. Solomon, Y.: The economics of mobile broadcast TV. Mobile DTV Alliance Whitepaper, Jan 2007 (2007)
9. Rauch, C., Kelleler, W.: Hybrid mobile interactive services combining DVB-T and GPRS. In: Proceedings of the European Personal Mobile Communications Conference (EPMCC) 2001, Vienna (2001)
10. Zhang, Y., Zhang, C., Cosmas, J., Loo, K.K., et al.: Analysis of DVB-H network coverage with the application of transmit diversity. IEEE Trans. Broadcast. **54**(3), 568–577 (2008)
11. G′omez-Barquero, D., Bria, A.: Forward error correction for file delivery in DVB-H. In: Proceedings of IEEE VTC Spring, Dublin (2007)
12. Lee, B.D., Song, J., Nam, Y.K.: Converged mobile TV services supporting rich media in cellular and DVB-H systems. IEEE Trans. Consum. Electron. **54**(3), 1091–1097 (2008)
13. Gardikis, G., Kormentzas, G., Xilouris, G., Koumaras, H., Kourtis, A.: Broadband data access over hybrid DVB-T networks.

In: Proceedings of the 3rd Conference on Heterogeneous Networks (HETNETs) '05, Ilkley (2005)
14. Gómez-Barquero, D., Bria, A.: Error repair for broadcast transmissions in DVB-H systems. Wirel. Commun. Mob. Comput. **9**(6), 733–744 (2009)
15. ETSI EN 302 304: Digital video broadcasting (DVB): transmission system for handheld terminals (DVB-H), European standard, v.1.1.1 (2004)
16. ETSI TR 102 401: Digital video broadcasting (DVB): transmission to handheld terminal (DVB-H), Validation Task Force Report (2005)
17. ETSI TR 102 377 v1.3.1: Digital video broadcasting (DVB): DVB-H implementation guidelines (2007)
18. ETSI: Digital video broadcasting (DVB): framing structure, channel coding and modulation for digital terrestrial television. ETSI EN300 744 v1.4.1 (2001)
19. ETSI TS 102 472 v1.2.1: IP Datacast over DVB-H: content delivery protocols (2006)
20. Kornfeld, M., May, G.: DVB-H and IP Datacast—broadcast to handheld devices. IEEE Trans. Broadcast. **53**(1), 161–170 (2007)