A Real-Time PC-Based Software Radio DVB-T Receiver

Shu-Ming Tseng Jian-Cheng Yu Yueh-Teng Hsu
Department of Electronic Engineering
National Taipei University of Technology
1, Sec. 3, Chung-hsiao E. Rd., Taipei,10608,Taiwan, R.O.C.
Taipei, Taiwan
shuming@ntut.edu.tw http://www.cc.ntut.edu.tw/~shuming/

Abstract: - We propose a PC-based real-time software Digital Video Broadcasting-Terrestrial (DVB-T) receiver, including hardware, driver, and software parts. The total data rate of DVB-T is much greater than that of Digital Audio Broadcasting (DAB). A more efficient signal processing algorithm than our previous software DAB receiver is necessary to improve the speed of the software part of the DVB-T receiver. Therefore, we propose some modifications in software: Viterbi algorithm optimization, and Digital-Down-Converter (DDC) optimization. Finally, it only takes 1541 ms to decode the 2760ms video data in this system, and the data recoded at 100km/hr vehicle speed is successfully decoded.

Key-Words: - software radio, mobility, Digital Video Broadcasting, orthogonal frequency-division multiplexing

1 Introduction

The most important benefit of Software Radio (SR) research, when compared with hardware radio, is that people can modify and change the signal processing procedure, the algorithm, and the result can be easily tested. Hence, more efficient detection or estimation algorithms are proposed with software language to improve the performance of a radio receiver rather than producing a new ASIC chip. The PC-based SR example includes a software Global Position System (GPS) [1] and software Digital Audio Broadcasting (DAB) capability [2]. The SR receivers, which are PC-based, are more suitable for research and as a development platform than the conventional hardware.

The total data rate of DAB is 2.4 Mbps (DQPSK); Digital Video Broadcasting-Terrestrial (DVB-T) is 9.953 Mbps (16 QAM, guard interval 1/4, rate 2/3 convolutional code, non-hierarchical system for 6 MHz channels) [3]. The data rate of DVB-T is much greater than the data rate of DAB. The software Viterbi decoder needs to be implemented faster due to much higher data rate than the DAB system [4]. The Reed-Solomon (RS) code is an additional feature in DVB-T, and already optimized in another journal paper [5]. The performance of the RS and Viterbi decoder is improved by several methods: (1) use the new SIMD instructions; (2) modify the program to reduce the branch; and (3) have more efficient and useful procedures.

In addition, we also propose new Digital Down Converter (DDC) to combine the mixer and filter operations to enhance the decoding speed in the software DVB-T receiver. We also implemented the hardware and driver necessary to construct a complete software DVB-T receiver.

This paper is organized as follows. First we describe the previous SR in Section 1. The system model block diagram is introduced in Section 2. The proposed system hardware is described in detail in Section 3. The driver is described in Section 4. The software is described in Section 5. Implementation and results are described in Section 6. Section 7 presents the conclusions.

2 Introduction

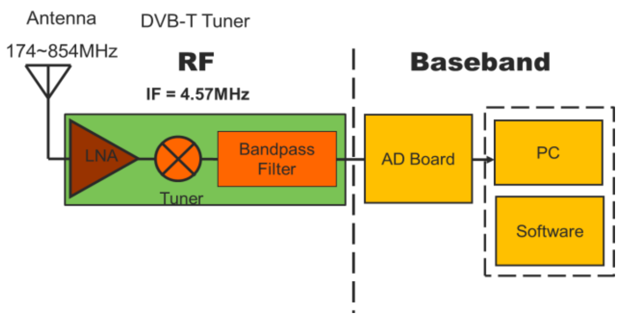

Fig. 1 System model block diagram of the introduced research platform

The proposed research architecture is shown in Fig. 1. The hardware part is depicted in Fig. 2. The receiver can decode radio signals and record baseband signals into the hard disk continuously.

The A/D chip and Low Voltage Differential Signaling (LVDS) are used to reduce the transmission interference. The architecture is shown in Fig. 3. The tuner module must have proper receiving sensitivity and low Intermediate Frequency (IF) signal output to amplify gain that is suitable for the requirement of an Analog-to-Digital (A/D) converter and so on.

3 Introduction

In our design, the IF power level must be in the A/D 8-bits sampling rate range. Once it is lower than the A/D sampling rate range, the resolution performance will be poor. On the contrary, if the IF power level is greater than the A/D sampling rate range, then it will generate distortion in the system. The IF power should be stable in a fixed range to prevent the A/D sampling rate shift. Additionally, it is necessary to have real-time Auto Gain Control (AGC). We prefer the tuner with the necessary amplifier feedback and self-control.

Thus, the silicon tuner (Xceive xc3028) is chosen to meet the requirements. The TLI 5540 is a high-frequency signal processing IC and converts the received frequency to IF 4.571 (MHz). The frequency command is fed through the I-squared-C (I2C) interface on this tuner. It is known that the sampling rate in the DVB-T is 6 MHz at 64 (MHz)/7 x 6/8 = 48 (MHz)/7 = 6.857 (MHz). We use 48 (MHz)/7 x 4 = 27.42857 (MHz) for 8-bit resolution sampling rates to be digitalized by an A/D converter. This is fed into a USB 2.0 chip and can be sent to aPC for future processing.

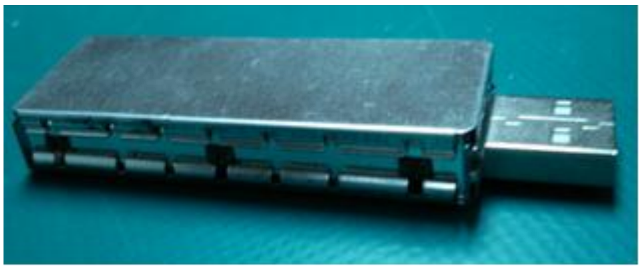

Fig. 2 USB interface; hardware part

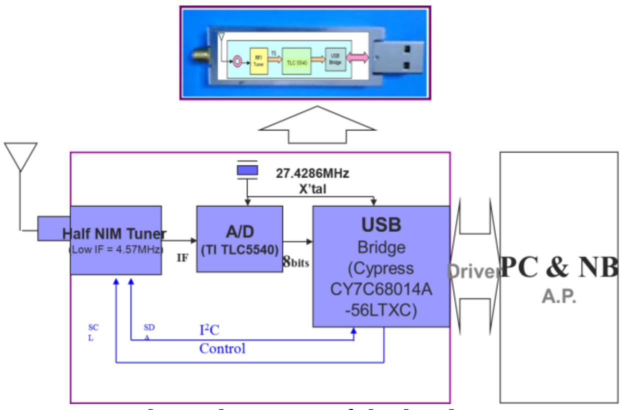

Fig. 3 The architecture of the hardware part.

4 Driver

It is necessary to develop a USB device driver in order to control this RF front end. In this research, the driver framework is a Windows Driver Module (WDM). It consists of the port driver, usbport.sys, and one or more of three miniport drivers that run simultaneously. Above the port driver is the USB bus driver, usbhub.sys, also known as the hub driver. The host controller driver and bus driver are both system-supplied USB drivers. They are utilized by our developed client driver. The relation between the three drivers is shown in Fig. 4.

The client driver's functions include collecting the digitized baseband data, notifying the user space application when the baseband date arrives, and setting the tuning frequency. The USB 2.0 high-speed mode (480 Mbits/second) was chosen for transferring. Since the baseband data is serially fed into the PC, to reduce the overhead of the data transfer, the isochronous mode is adopted.

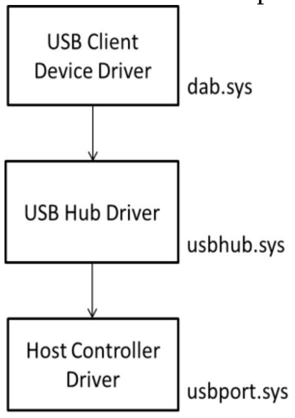

Fig. 4 USB driver stack

The USB client device driver communicates with the USB hub device driver with a USB Request Block (URB. Its architecture is shown in Fig. 4. The digitalized baseband data is fed into one of the double buffers. When collected data reaches the threshold, the double buffer will switch and send a message event to notify the application in user space. Once the application receives the event, it copies the data from the driver and performs signal processing continuously.

5 Software

The software has two parts: one is a baseband receiver, and another is a baseband data recorder. The signal processing of the software structure is shown in Fig. 5. and Fig. 6. The time synchronization processis used to synchronize the symbol in the Fast Fourier Transform (FFT) window correctly. We use the Continual Pilot (CP) to estimate integral frequency offset and use Transmission Parameter Signaling (TPS) pilot to estimate the offset of the Orthogonal Frequency-Division Multiplexing (OFDM) symbol. Fractional frequency synchronization can modify the signal after the digital down converter is in the correct baseband frequency.

We will use a scatter plot to estimate the channel coefficients. After doing fractional frequency synchronization and integer frequency synchronization, the residual frequency offset still exists. However, we can utilize a mathematic model for an offset with sampling frequency offset and residual frequency. Channel estimation must be performed to compensate for the phase error. The TPS is used to do frame synchronization in theDVB-T system.

We describe the RS decoder optimization in Section 5.1. The Viterbi decoder and the systematic approach of software optimization is detailed in Section 5.2. DDC design and optimization is described in Section 5.3.

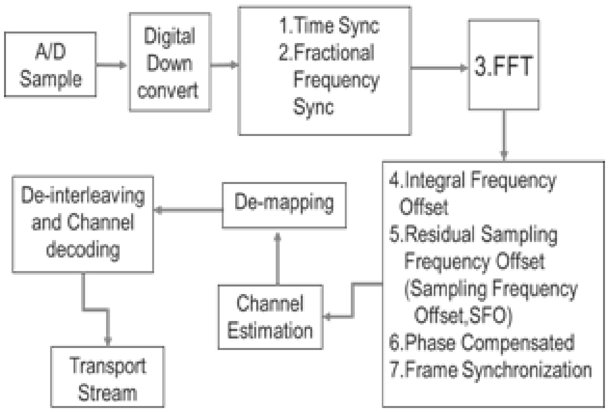
A/D Sample
Digital Down convert
1.Time Sync
2.Fractional Frequency Sync
3.FFT
4.Integral Frequency Offset
5.Residual Sampling Frequency Offset (Sampling Frequency Offset,SFO)
6.Phase Compensated
7.Frame Synchronization
Channel Estimation
De-mapping
De-interleaving and Channel decoding
Transport Stream

Fig. 5 Structure of software

5.1 Reed-Solomon decoder optimization

The RS decoder is the most time-consuming part of the software DVB-T receiver. We must improve the performance of the RS decoder, which is composed of four parts. In this subsection, we briefly describe the our previous RS decoder optimization in [5]. Each part uses different efficient algorithms that are described as follows. First, there are 4064 addition and 4080 multiplication actions in GF (256) for the getting syndrome:

These operations can be replaced by the proposed 255 vector tables; each table means that all possible answers of 16 multiplications in GF (256) are obtained. Hence, you can get 16 multiplications in GF (256) by looking up the look-up vector table once. The multiplication of GF (256) is no longer necessary. The GF (256) is created with look-up vector tables (shown as Table 1). Furthermore, we rewrite the whole Chien search program in an assembly language.

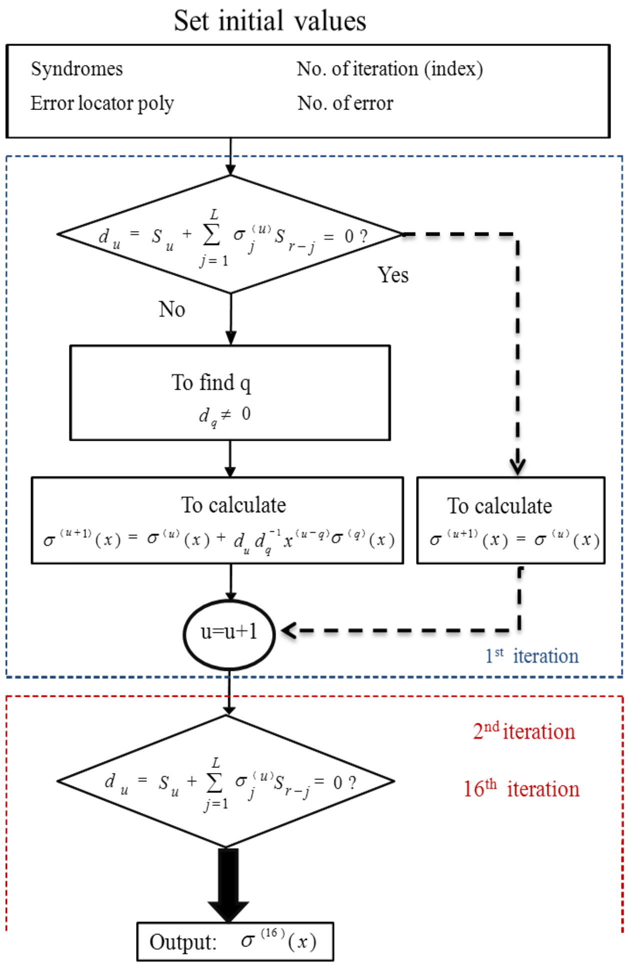
Set initial values
Syndromes
No. of iteration (index)
Error locator poly
No. of error
$d_u = S_u + \sum_{j=1}^{L} \sigma_j^{(u)} S_{r-j} = 0$?
Yes
No
To find q
$d_q \neq 0$
To calculate
$\sigma^{(u+1)}(x) = \sigma^{(u)}(x) + d_u d_q^{-1} x^{(u-q)} \sigma^{(q)}(x)$
To calculate
$\sigma^{(u+1)}(x) = \sigma^{(u)}(x)$
u=u+1
1st iteration
2nd iteration
16th iteration
$d_u = S_u + \sum_{j=1}^{L} \sigma_j^{(u)} S_{r-j} = 0$?
Output: $\sigma^{(16)}(x)$

Fig. 6 Modified procedure of Berlekamp-Massey (BM) in our program The performance of the Chien search program will be much better by virtue of these modifications. Finally, the decoding speed is much faster than the previous Chien search program.

Table 1. Second Vector Table

r_1	$r_1\alpha^0$	…	$r_1\alpha^{15}$
1	1	…	38
⋮	⋮	⋮	⋮
255	255	…	174

The branch commands like IF, FOR, WHILE, etc., have prediction errors that will cause the speed of the execution to slow down. It is necessary to reduce all the loops in our program by means of using a large number of C or assembly codes. Thus, we rewrite some programs to expand those C or assembly codes, and the new procedure is shown in Fig. 6. Sixteen XMM registers are provided by the 64-bit CPU. It is helpful and useful to place all operational values of Forney algorithm procedures using these registers. It can save time to transfer the data between registers and memories as much as possible. The data are only moved between the registers.

5.2 Viterbi decoder and systematic approach of software optimization

We use Explorer in the Microsoft Visual Studio 2008 (Team Suite edition) to identify and analyze the slowest parts in our programs. Respectively, we propose several ways to improve performance. The Viterbi decoder consists of several units: Branch Metric Unit (BMU), Add-Compare-Select Unit (ACSU), Path Metric Memory Unit (PMMU), and Survivor Memory Unit (SMU). We use the "compare instructions" of the Streaming SIMD Extension 2 (SSE2) application several times to determine the minimum and we use its index in the XMM register for tracing back. Furthermore, the new instructions of the SSE4 for the Intel® CPU ("PHMINPOSUW") are used to find the minimum and the index of the minimum (as shown in Fig. 7). We use these instructions to determine the decoding path in the Viterbi algorithm. There are many loops in our original Viterbi decoder C program. In order to reduce the loops in our program, we write some programs to automatically generate duplicate C or assembly code segments. For example, there is a loop repeating 64 times in our original 64-state Viterbi decoder, and we write a program to produce a large amount of C code (Fig. 8) in which the program segment inside the loop is repeated 64 times.

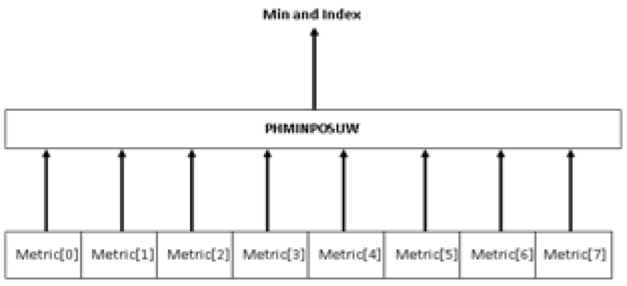

Fig. 7 Finding the minimum and index with SSE2+SSE4.

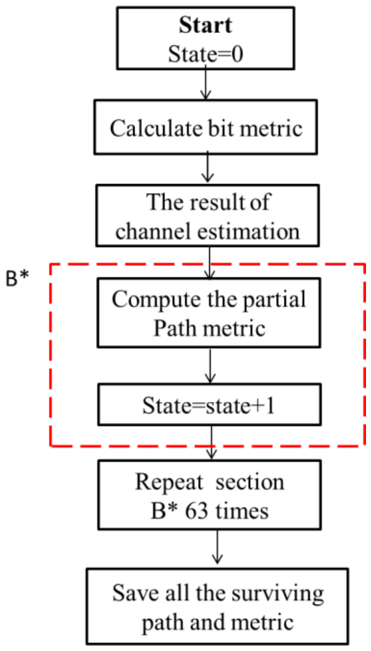
Start
State=0
Calculate bit metric
The result of channel estimation
B*
Compute the partial Path metric
State=state+1
Repeat section B* 63 times
Save all the surviving path and metric

Fig. 8 The flowchart of the modified program. A loop isexpanded to 64 identical segments b*

5.3 Viterbi decoder and systematic approach of software optimization

The DDC is an important part of DVB-T system; it converts the IF signal into baseband, reduces the signal sampling rate, and then makes it easy for realtime demodulation. The structure of the previous DDC is comprised of the mixer, low-pass FIR, and down-sampling given by:

$$x(n) = s(n) * \exp(-j2\pi * f_{IF} * (n / f_{AD})), \quad 1 \le n \le N \tag{1}$$

where N is the length of DDC input data, fIF and fAD represent the IF and sampling rate of A/D.

We propose a Combining the Mixer and Filter (CMF) method. Respectively, we multiply the mixer coefficients and filter coefficients and store the results in a look-up table. In [6], fAD is four times as much as fIF; it could simplify the calculation of the mixer. Furthermore, integer decimation is proposed in [7]. According to [6] and [7], changing the fAD to be a multiple of fIF and the sampling rate of DDC output would simplify the DDC computation. Hence, we choose the fAD to be 192/7 MHz. For this sampling rate, it will match the multiple relation in [6] and [7] at the same time. The DDC with fAD = 192/7 MHz is shown in Fig. 9.

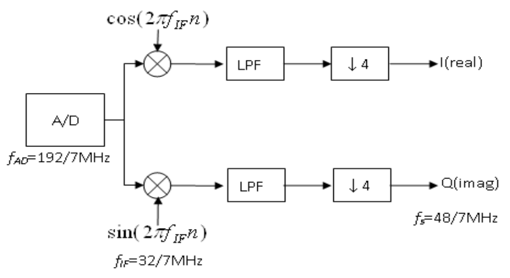

Fig.9 DDC using fAD=192/7 MHz

This architecture avoids up-sampling calculation. Thus, the structure of fAD = 192/7 MHz is simpler than the structure of fAD = 20 MHz, so we choose 192/7 MHz as fAD in our structure.

f_{IF} / f_{AD} = 32/7 ÷ 192/7 = 1/6 in (1), we have:

$$x(n) = s(n) * \exp(-j2\pi * (n*1/6)), \quad 0 \le n \le N-1 \tag{2}$$

We note that we only have six possible values. The filter h(m) only has M+1 coefficients.

$$k(n) = \sum_{m=0}^{M} \{s(n-m) * [w((n-m) \bmod 6) * h(m)]\}, \quad 0 \le n \le N-1 \tag{3}$$

where we define w(n) = In order to save the elapsed time from mixer calculation, we modify (3) to be:

$$k(n) = \sum_{m=0}^{M} \{s(n-m) * [w((n-m) \bmod 6) * h(m)]\}$$
$$= \sum_{m=0}^{M} \{s(n-m) * C(m)\}, \quad 0 \le n \le N-1 \tag{4}$$

The w(n) and h(m) are combined in advance in (4). In addition, we calculate the linear convolution between s(n) and c(m) to achieve the mixer and filter calculations. The block diagram of the proposed CMF algorithm is shown in Fig. 10.

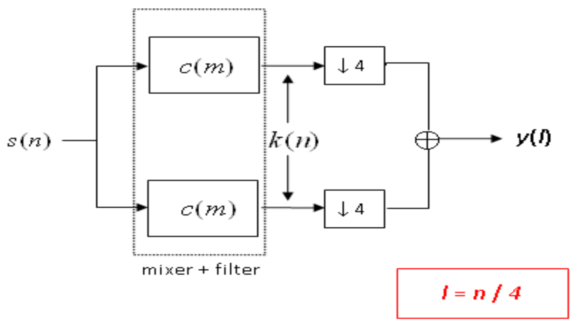

Fig. 10 Block diagram of the proposed CMF

The next step is to down sample four times. Down-sampling is usually combined with (4). That is, when we input four data points to the DDC, the output is one data point. This procedure could avoid 3/4 calculations from (4). When the down-sampling is combined, we get:

$$y(l) = k(n) = \sum_{m=0}^{M} \{s(n-m) * [w((n-m) \bmod 6) * h(m)]\}$$
$$= \sum_{m=0}^{M} s(s-m) * c(m), n = 3,7,11...., N-1 \qquad l = (n-3)/4 \tag{5}$$

Assume the number of the DDC input data is N; there will be N multiplications in (2). Furthermore, our FIR filter is seven orders, so there are 8N multiplications generated by linear convolution of (4). In fact, down-sampling is usually combined with the FIR. According to this, the 8N multiplications will reduce to 2N because of down-sampling: four times in (5). To sum up, the number of multiplications in the previous DDC [8] is 3N (=N+2N). The CMF combines mixer and filter, so the number of multiplications could be reduced to 2N additionally. Hence, the CMF can save more elapsed time than the previous DDC algorithm.

The optimization result of our DDC algorithm is shown in Table 2. We choose Windows 7 (64 bits) as our Operating System (OS) in order to use all 16 XMM registers. Furthermore, the development tool is Microsoft® Visual Studio 2010 (Team Suite edition). The elapsed time of the proposed DDC can be shown via Performance Explorer in assembly and C codes.

Table 2 DDC elapsed times of fAD =192/7 MHz (in C and assembly

Function name	**Elapsed time (ms)**
ddc (C)	88.07
ddc_asm (assembly)	20.81

6 Implementation and results

The environment of the proposed receiver is a desktop PC whose specification is listed in Table 3. The OS is Windows 7. The software language is C, C++, or assembly.

The programming tools are mainly from Microsoft® and Intel including the compiler and mathematical library.

Table 3 PC platform specification including OS

Component	Spec
OS	Windows 7 (64-bit)
CPU	Intel® (R) Core (TM) i7-2600 K CPU @ 3.40 GHz (8 CPUs), ~3.4 GHz
RAM	4.0 GB
Motherboard	ASUS P8H67-M PRO

	(REV3.0)

In this research, the CPU loading of each individual block is listed in Table 4. The final results are shown as Fig. 11.

Table 4 CPU loading of each block for 2760 ms video data

Block	Elapsed Inclusive Time (ms)
Time and frequency synchronization	63.66
Remove CP and FFT	67.82
Channel estimation	145.87
Deinner and depuncher	54.49
Deoutter interleave	17.93
Demodulator	8.93
Viterbi decoder	1096.79
RS decoder	30.67
Descrambler	0.92
Frame synchronize	8.58
Program initialization	27.52
Phase compensation	8.44
C++ standard library	10.27
Total	1541

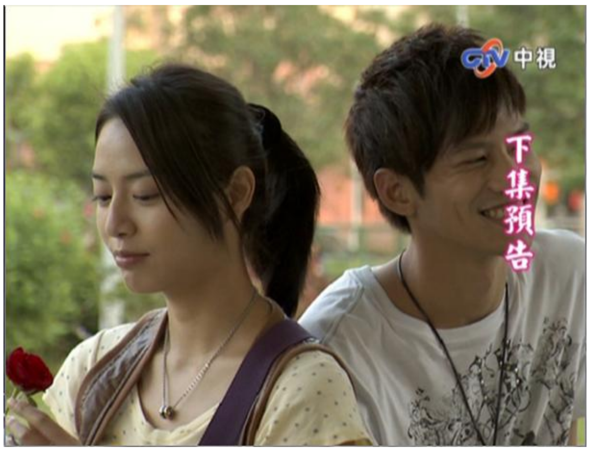

Fig. 11 Software demodulator results

7 Conclusion

In this paper, we utilize some efficient methods to greatly improve the decoding speed of our Viterbi decoder. Furthermore, the new DDC can also be used to enhance the performance of the SR receiver. As a result, the decoding rate of the proposed system is greater than the required bit rate of the real-time DVB-T. So, our system is fast enough to decode the DVB-T signal in real-time.

The proposed system is convenient to operate with the current commercial PCs and it provides a platform to develop new baseband algorithms. We have verifed it in a real-world 100km/hr environment. Finally, it only takes 1541 ms to decode the 2760 ms of video data. Thus we have implemebented a real-time software DVB-T receiver.

References:

[1] N. Kubo, S. Kondo, and A. Yasuda, “Evaluation of code multipath mitigation using a software GPS receiver,” IEICE Trans. Commun., Vol.E88-B , No.11, Nov. 2005, pp. 4204 -4211.

[2] Shu-Ming Tseng, Yueh-Teng Hsu, Meng-Chou Chang, and Hsiao-Lung CHAN, “A Notebook PC Based Real-Time Software Radio DAB Receiver,” IEICE Trans. Commun., Vol. E89-B, No. 12, Dec. 2006, pp. 3208-3214.

[3] ETSI EN 300 744 :Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television ETSI, 2004.

[4] Shu-Ming Tseng,Yu-Chin Kuo,Yen-Chih Ku, and Yueh-Teng Hsu, “Software Viterbi Decoder with SSE4 Parallel Processing Instructions for Software DVB-T Receiver, ” in Proc. The 7th IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA-09), Aug. 2009, pp. 102-105.

[5] Shu-Ming Tseng, Yueh-Teng Hsu, and Jheng-Zong Shih, "Reed-SolomonDecoder Optimization for PC-Based DVB-T Software Radio Receiver, "Information -An International Interdisciplinary Journal, accepted.

[6] Ji-yang Yu, and Yang Li, “An Efficient Digital Down Converter Architecture for Wide Band Radar Receiver,” in Proc 2009 IET International Radar Conf., April 2009, pp. 1-4.

[7] M. J. Zhao, P. L. Qiu and J. H. Tang, “Sampling rate conversion and symbol timing for OFDM software receiver,” IEEE 2002 International conference on Communications, Circuits and Systems and West Sino Expositions, Vol. 1, May, pp. 114-118.

[8] Yih-Min Chen, “On the Design of Farrow Interpolator for OFDM Receivers with Asynchronous IF Sampling,” Fourth International Conference on Communications and Networking in China, Aug. 2009, pp. 1-5.

[9] S. B. Wicker, Error Control Systems for Digital Communication and Storage, Prentice Hall, 1995.