Software Digital-Down-Converter Design and Optimization for DVB-T Systems

Shu-Ming Tseng
Graduate Institute of Computer and Communication Engineering
National Taipei University of Technology
Taipei, Taiwan
shuming@ntut.edu.tw

Jian-Cheng Yu
Graduate Institute of Computer and Communication Engineering
National Taipei University of Technology
Taipei, Taiwan
g931303@hotmail.com

Zheng-Hau Lin
Graduate Institute of Computer and Communication Engineering
National Taipei University of Technology
Taipei, Taiwan
steve76813@hotmail.com

***Abstract* –In this paper, a novel Digital-Down-Converter (DDC) architecture for PC-based software Digital Video Broadcasting-Terrestrial (DVB-T) receiver is proposed. The sampling rate of A/D is 192/7 MHz and the order of the lowpass FIR filter is seven (8 coefficients) in DDC. Furthermore, the Combining of the Mixer and Filter (CMF) is also proposed, including the pre-processing of mixer and filter coefficients and storing the results in a look-up table. If the input data of length is N, the proposed CMF scheme has 2N multiplications, while the previous architecture, which does not pre-process the mixer and filter coefficient together in advance, has 3N multiplications. Finally, the algorithms also are optimized in assembly code to satisfy DVB-T real-time reception requirement.**

Keywords- DVB-T; DDC; Decimate; Real-time.

I. INTRODUCTION

The structure of the receiver end of Digital Video Broadcasting –Terrestrial (DVB-T) system [1] is shown in Figure 1. The Digital Down Converter (DDC) is an important part of DVB-T system; it converts the Intermediate Frequency (IF) signal into baseband, reduces the signal sampling rate and then makes it easy for the later real time demodulation. The traditional DDC in DVB-T system was made by hardware circuits [2] [3]. However, it is not easy to be integrated and modified. The most important benefit of Software Radio (SR) research is that people can modify and change the signal processing procedure, the algorithm, and the result can be easily tested. Hence, there are some researches implemented DDC by software [4] [5].

In Figure 1, the IF generated by the tuner is 32/7MHz, defined by the DVB-T standard [1]. Besides, the sampling rate of DDC output in DVB-T standard must be 48/7MHz [1]. The previous DDC structure is shown in Figure 2, it composes of mixer 、lowpass Finite Impulse Response (FIR) filter and down-sampling.

A Combining the Mixer and Filter (*CMF)* method is proposed. In other words, the mixer coefficients and filter coefficients are multiplied and the results are stored in a look-up table, as shown in (5). Furthermore, The *CMF* algorithms also are optimized in assembly code. The proposed *CMF* scheme combines the mixer and filter operations. Hence, it can reduce the computational complexity by one third. Assume the input data of length N, the previous scheme has 3*N* multiplications, but the proposed *CMF* only has 2*N*.

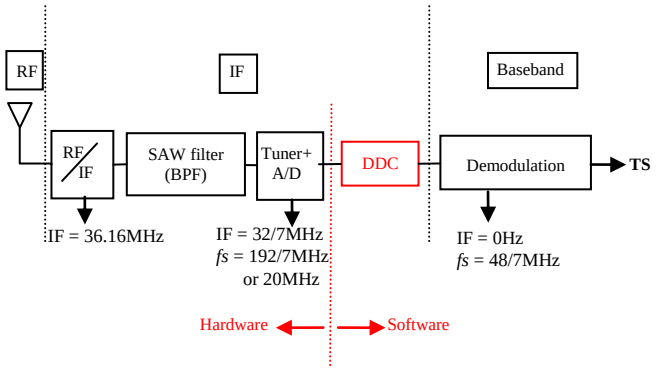

Figure 1. The structure of DVB-T receiver

The rest of this paper is organized as follows: In Section II, the system model of DDC is presented. The decision of A/D sampling rate is described in Section III. The proposed CMF scheme for DDC is described in details in Section IV. The CMF optimization is presented in Section V. Performance discussion is described in Section VI. Finally, we conclude in Section VII.

II. SYSTEM MODEL

The previous of DDC is introduced in Figure 2. According to [6], received signal s *(n)* is transformed to baseband by mixer first:

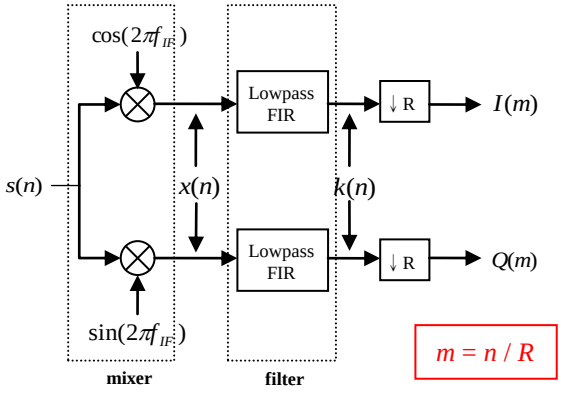

Figure 2. The DDC of previous structure

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

where N is the length of DDC input data, f_{IF} and f_{AD} is IF and sampling rate of A/D, respectively. The IF power level must be in the sampling rate range. Once it is lower than the A/D

 ISBN: 978-1-61208-192-2

sampling rate range, the resolution performance will be poor. On the contrary, if the IF power level is higher than the A/D sampling rate range, then it will produce distortion in the system. And then employ a low-pass FIR to avoid aliasing effect after down-sampling:

$$k(n) = h(0)x(n) + h(1)x(n-1) + \cdots + h(M)x(n-M)$$
$$= \sum_{m=0}^{M} h(m)x(n-m) = h(n) \otimes x(n) \quad (2)$$

where M is the order of FIR filter, $h(n)$ is the filter coefficients, and "$\otimes$" is linear convolution. Final step is down-sampling; R means the down-sampling rate. After down-sampling, the data stream will satisfy the required sampling rate of DVB-T standard.

III. CHOICE OF A/D SAMPLING RATE

In Figure 3, most of f_{AD} is following the commercial specification: 20MHz [5] [7] to record data. According to DVB-T standard [1], the IF of tuner is 32/7MHz. Besides, the sampling rate of DDC output in DVB-T standard must be 48/7MHz [1].

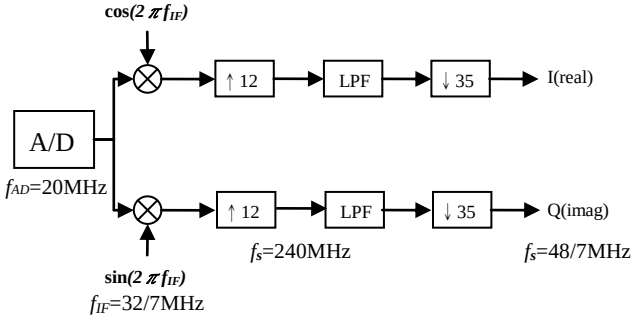

Figure 3. Digital-Down-Converter in [5] [10]

Because these three (20, 32/7, 48/7) are not in a multiple relationship, it will increase computational complexity.

In [8], f_{AD} is 4 times as much as IF; it could simplify the calculation of mixer. Moreover, integer decimation is proposed in [9]. According to [8] [9], changing the f_{AD} to be multiple of IF or the sampling rate of DDC output would simplify the DDC computation. Hence, the 192/7MHz of f_{AD} is chosen. It will match the multiple relation in [8] [9] simultaneously. For architecture of f_{AD} is 192/7MHz, as shown in Figure 4. This architecture avoids up-sampling calculation compared with Figure 3. The structure of f_{AD} = 192/7MHz is simpler than f_{AD} = 20MHz's. Thus, the 192/7MHz is chosen as f_{AD} in our structure.

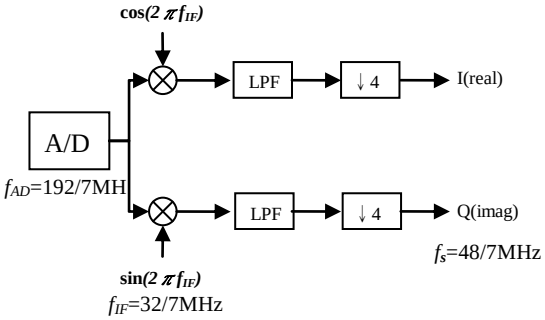

Figure 4. Digital-Down-Converter using f_s=192/7MHz

The environment of the hardware whose specification is listed in Table I. As we know, the case A has lots of additional up-sampling computations. From Table II, the case B can save much more time than case A.

TABLE I. HARDWARE LIST

Item	Model
CPU	Intel® Core™ i7-2600K (3.40GHz)
Memory	DDR2 800 2GB x 2
Main board	ASUS P8H67-M PRO Rev 1.xx
Graphic card	n.a

TABLE II. THE ELAPSED TIMES OF DIFFERENT F_{AD} (IN MATLAB)

	Elapsed time(s)
A. f_{AD} = 20MHz	441.28
B. f_{AD} = 192/7MHz	175.55

IV. PROPOSED CMF SCHEME

In this section, the new *CMF* scheme is proposed to simplify the DDC computation. The architecture without proposed CMF scheme is shown in Figure 4. Because f_{IF} / f_{AD} = 32/7 ÷ 192/7 = 1/6, we have:

$$x(n) = s(n) * \exp(-j2\pi * (n * 1/6)), \quad 1 \le n \le N \quad (3)$$

From (3), the $\exp(-j2\pi*(n*1/6))$ only have possible 6 values. Furthermore, the filter only has M+1 coefficient. Substitute (3) into (2) and we get:

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

where $w(n)$ = $\exp(-j2\pi*(n*1/6))$ is defined. In order to save the elapsed time from mixer calculation, the formula (4) will be modified as below:

$$k(n)=\sum_{m=0}^{M}\{s(n-m)*[w((n-m)\bmod 6)*h(m)]\},\quad 0\le n\le N-1$$
$$=\sum_{m=0}^{M}\{s(n-m)*c(m)\},\quad 0\le n\le N-1 \qquad (5)$$

The major contribution of this paper is combining the w (n) and h (m) in (5) is a look-up table in advance, then s (n) calculates the linear convolution with c (m) will achieve the mixer and filter calculations. Hence, it can save the time for computation. Because the number of mixer coefficients is not equal to filter's, we choose the least common multiple of these two numbers: 24. Figure 5 shows the combination of look-up table between mixer and filter. Here we assume M = 7. We also set M = 7 in our architecture, the detail reason will be described in next section.

Observing Figure 5, The 4 groups of mixer coefficients and 3 groups of filter coefficients are used to form the look-up table. The look-up table will be divided into three parts: ①、② and ③, so the received signal s(n) operate (5) with ①、② and ③ circularly. The next step is down-sampling 4 times. This part could be combined with (5). For each 4 data input in DDC, it will generate 1 data. Thus, It could avoid 3/4 calculations form FIR.

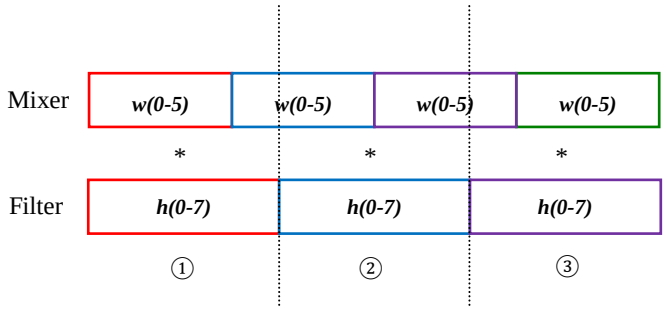

Figure 5. Look-up table of mixer and filter combination

According to (5), the block diagram of proposed algorithm: *CMF* is shown in Figure 6.

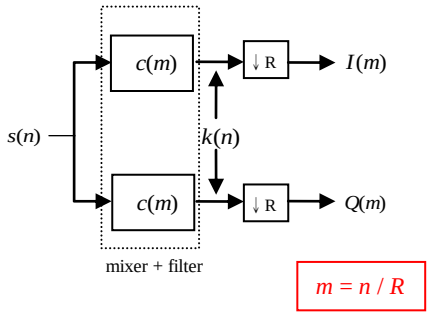

Figure 6. The block diagram of proposed *CMF*

V. *CMF* OPTIMIZATION AND IMPLEMENT IN ASSEMBLY

A. Choice of filter order

The architecture is implemented on a personal computer by software; the most important goal is fast enough to process the data. As we know, the XMM register is formed 128 bits. Hence, it could contain 16 signed byte data. In order to utilize the XMM registers efficiently, 7 orders FIR is chosen. In other words, there are 8 filter coefficients. The data type of designed filter coefficients is float. The look-up table coefficients to be integer which data type is byte data. The two groups of look-up table coefficients could be built in a XMM register. Thus, we achieve two linear convolutions in a XMM register. For example, a filter instance design for DDC in [10] is 9 orders (10 coefficients); a XMM register can only hold one group of filter coefficients. As a result, it can reduce the computational complexity.

B. Assembly implement

The parallel processing instructions is used in assembly and XMM registers. The assembly computation is mainly between the XMM registers. Besides, the new DDC calculation is based on the linear convolution in (5). Hence, the two Supplemental Streaming SIMD Extensions 3 (SSSE3) instructions are chosen to achieve PMADDUBSW and PHADDSW [11] which described as below.

PMADDUBSW: Multiply and Add Packed Signed and Unsigned Bytes.

PHADDSW: Packed Horizontal Add and Saturate Words.

These two instructions are shown in Figure 7 and Figure 8:

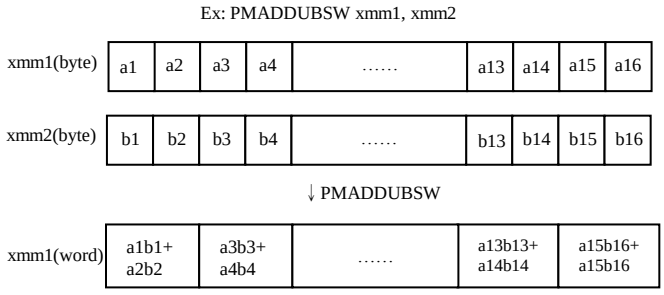

Figure 7. SIMD instruction: PMADDUBSW

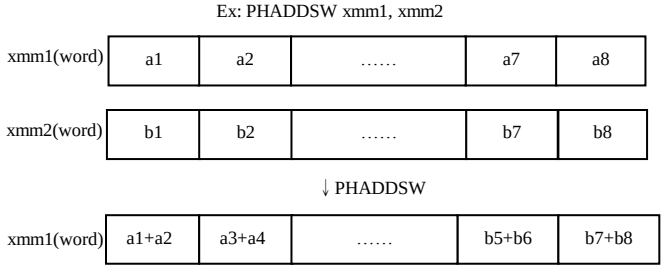

Figure 8. SIMD instruction: PHADDSW

The "PMADDUBSW" is used to multiply *s* (*n*) and *c* (*m*) in (5) together. The "PHADDSW" is used twice to sum the results in all. The data type of *s* (*n*) and *c* (*m*) in (5) is unsigned byte and signed byte, respectively. Because of the data type, only "PMADDUBSW" of SSSE3 instructions could achieve the multiplication between *s* (*n*) and *c* (*m*). The data type of "PMADDUBSW" outputs is signed word. Furthermore, the "PHADDSW" of SSSE3 instructions are used to add the outputs horizontally. The detail linear convolution example achieved by these two instructions is shown in Figure 9.

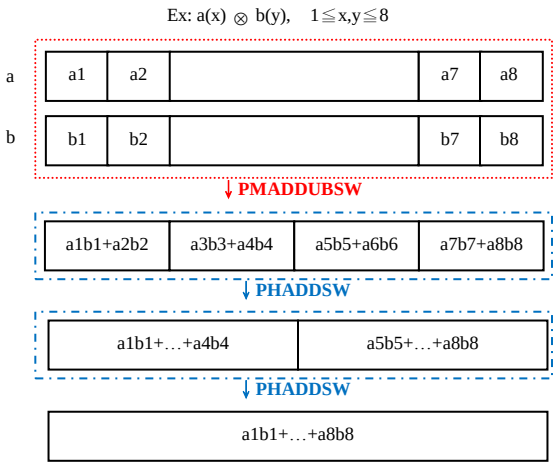

Figure 9. Linear convolution by "PMADDUBSW","PHADDSW"

VI. PERFORMANCE OF OUR DDC

The new DDC algorithm will be compared with the previous architecture. First, the DDC calculation is presented by Matlab. Then, we could know the improvement of *CMF* method. The quantity of input data is 84Mbytes; Table I is our hardware simulation environment.

Table III presents the elapsed time about f_{AD} = 192/7MHz use *CMF* or not. The mixer and filter coefficient is multiplied in advance; DDC algorithm could be simpler in (5). In our case, if the number of DDC input data is *N*, there will be *N* multiplications in (3). Moreover, our FIR filter is 7 orders, so there has 8*N* multiplications generated by linear convolution of (5). In fact, the down-sampling part usually combines with the FIR. The 8*N* multiplications will reduce to 2*N* because of down-sampling: 4 times. The number of multiplications in the previous DDC is 3N (=N+2N). The CMF combines mixer and filter, so the number of multiplications could be reduced to 2N additionally. Thus, the CMF can save much more elapsed time than the previous DDC algorithm. As a result, the new DDC algorithm can save about 40% elapsed time.

TABLE III. THE ELAPSED TIMES OF F_{AD} =192/7MHZ USE *CMF* OR NOT (IN MATLAB)

	Elapsed time(s)	Multiplication
f_{AD} =192/7MHz (No)	175.55	3*N*
f_{AD} =192/7MHz (Yes)	104.89	2*N*

The optimization result of proposed DDC algorithm is shown in Table IV. In order to use all 16 XMM registers, the Windows 7 (64 bits) is chosen as system OS. Besides, the Microsoft Visual Studio 2010(Team Suite edition) is used as development tool. It has the Performance Explorer to analyze the elapsed time of proposed DDC in assembly code and C code.

TABLE IV. THE ELAPSED TIMES OF F_{AD} =192/7MHZ (IN C CODE AND ASSEMBLY CODE)

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

The DDC takes 20.81ms to decode 84Mbytes data. 84Mbytes/20.81ms = 4.04Gbytes/sec. According to f_{AD} = 192/7MHz, the real time DVB-T signal in Taiwan has 27Mbytes/sec. Hence, the proposed DDC only takes 0.67% CPU loading.

For the DVB-T software radio implement, the elapsed time of demodulation part is shown in Table V. The length of decoding data is 3 sec. However, the total elapsed time of demodulation part is 1541.89ms. Thus, the DVB-T real-time computation can be implemented (1.562sec < 3sec).

TABLE V. THE ELAPSED TIME OF DEMODULATION PART

Block	Elapsed Time (ms)
Time & Frequency Synchronize	63.66
Remove CP & FFT	67.82
Channel estimation	145.87
Deinner & 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 elapsed time	1541.89

VII. CONCLUSION

In this paper, the new DDC architecture is designed for f_{AD} = 192/7MHz and the algorithms are optimized by multiplying the

 ISBN: 978-1-61208-192-2

mixer and filter recorded in a look-up table in advance. The proposed *CMF* can save the additional *N* multiplications. Moreover, it is also optimized by assembly code. As a result, the decoding rate of the proposed system is greater than the required bit rate of the real-time DVB-T. The new system is fast enough to decode the DVB-T signal in real-time.

Finally, the new method can save more 40% time than the previous architecture. It only takes 20.81ms to decode 84Mbytes data. To sum up, the new DDC coding rate is 4.04Gbytes/sec.

ACKNOWLEDGMENT

The authors would like to thank all colleagues and students who contributed to this study.

REFERENCES

[1] Digital Video Broadcasting (DVB); Frame structure, channel coding and modulation for digital terrestrial television, European Standard (EN) 300 744 V1.5.1, European Telecommunications Standards Institute (ETSI), Nov. 2004.

[2] Mitchell, J. and Sadot, P., "Development of a digital terrestrial front end," International Broadcasting Convention 12-16 Sept. 1997, IEE Conference Publication No. 447, pp. 519-524, ISSN 0537-9989.

[3] Makowitz, R. Anikhindi, S. Gledhill, J. Mayr, M. and Drozd, M., "A Single-chip Front End for DVB-T Receivers," in Proc Consumer Electronics Con, Jun. 1998, pp. 390-391.

[4] F. Harris and R. W. Lowdermilk, "Software defined radio: Part 22 in a series of tutorials on instrumentation and measurement," IEEE Instrum. Meas. Mag., vol. 13, no. 1, pp. 23–32, Feb. 2010

[5] Fang-Hsu Lu and You, S.D., "Channel Estimation for DVB-T Receiver with Asynchronous Sampling Clock," IEEE 13th International Symposium on Consumer Electronics, May. 2009, pp.513-517.

[6] 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.

[7] Karim Medhat Nasr, John P. Cosmas, Maurice Bard, and Jeff Gledhill, "Performance of an Echo Canceller and Channel Estimator for On-Channel Repeater in DVB-T/H Networks," IEEE Trans. on Broadcasting, vol. 53, issue:3, pp. 609-618, Sep. 2007.

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

[9] 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, pp. 114-118, May. 2002.

[10] Yih-Min Chen, and I-Yuan Kuo, "Design of Lowpass Filter for Digital Down Converter in OFDM Receivers," International Conference on Wireless Networks, Comm. and Mobile Computing, vol. 2, Jun. 2005, pp. 1094-1099.

[11] "Intel® Advanced Vector Extensions Programming Reference," http://software.intel.com/file/19151

[12] "Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 1," http://download.intel.com/design/processor/manuals/253665.pdf

[13] "Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2A," http://download.intel.com/design/processor/manuals/253666.pdf

[14] "Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2B," http://download.intel.com/design/processor/manuals/253667.pdf

 ISBN: 978-1-61208-192-2