J Inf Process Syst, Vol.15, No.4, pp.890~903, August 2019
https://doi.org/10.3745/JIPS.04.0127

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

FPGA Implementation of SC-FDE Timing Synchronization Algorithm

Suyuan Ji*, Chao Chen*, and Yu Zhang**

Abstract

The single carrier frequency domain equalization (SC-FDE) technology is an important part of the broadband wireless access communication system, which can effectively combat the frequency selective fading in the wireless channel. In SC-FDE communication system, the accuracy of timing synchronization directly affects the performance of the SC-FDE system. In this paper, on the basis of Schmidl timing synchronization algorithm a timing synchronization algorithm suitable for FPGA (field programmable gate array) implementation is proposed. In the FPGA implementation of the timing synchronization algorithm, the sliding window accumulation, quantization processing and amplitude reduction techniques are adopted to reduce the complexity in the implementation of FPGA. The simulation results show that the algorithm can effectively realize the timing synchronization function under the condition of reducing computational complexity and hardware overhead.

Keywords

FPGA Implementation, SC-FDE, Timing Synchronization

1. Introduction

With the continuous development and progress of society, communication and communication have become an integral part of people's lives. Wireless communication technology is a communication method that utilizes the characteristics that electromagnetic signals can be freely transmitted in space to exchange information, it has the characteristics of convenient use, good scalability, and low cost. Wireless communication technology does not require a large-scale supporting infrastructure, has low installation and maintenance costs, and is highly adaptable to the environment. With the rapid development of science and technology, wireless communication technology has also been used initially for the exchange of user voice information, and gradually applied to cluster communication, satellite communications, mobile video technology and other aspects. But wireless communication also has some defects, such as weak anti-interference ability, slow transmission speed, limited bandwidth and limited transmission distance.

Single carrier frequency domain equalization (SC-FDE) is a kind of good antimultipath techniques. Compared with orthogonal frequency division multiplexing (OFDM), SC-FDE has the advantages of low

Manuscript received November 15, 2017; first revision February 22, 2018; second revision April 12, 2018; accepted May 1, 2018.
Corresponding Author: Chao Chen (chenchao@cuc.edu.cn)
* Engineering Center of Digital Audio and Video, Communication University of China, Beijing, China (1092037028@qq.com, chenchao@cuc.edu.cn)
** Communication and Technology Bureau, Xinhua News Agency, Beijing, China (zhangyu@xinhua.org)

peak to average power ratio and insensitivity to carrier phase. In recent years, SC-FDE has been widely concerned and applied in wireless communication systems [1]. In SC-FDE system, timing synchronization error will not only cause signal amplitude and phase distortion, but also cause inter-symbol interference, thus, seriously affecting the system performance. Therefore, timing synchronization has always been the focus of research and directly affects SC-FDE system performance. Because of the rapid development of digital processing chips such as field programmable gate array (FPGA) and digital signal processing (DSP), the hardware realization of complex signal processing algorithms has become a reality. Therefore, realizing SC-FDE timing synchronization algorithm on FPGA has strong engineering significance.

The SC-FDE timing synchronization has been studied in a large amount of literature, it can be divided into three main categories: synchronous algorithm based on cyclic prefix (CP), synchronous algorithm based on special structure, and synchronization algorithm based on training sequence. In [2], because of the use of CP, there is a plateau area, the precision is not high, and it cannot even work under the influence of the multipath channel. The authors [3] use the particularity of the conjugate symmetric structure for timing synchronization, it performs poorly in low signal-to-noise ratio (SNR) and is more complex in the FPGA implementation. In [4], the authors use good autocorrelation and cross-correlation of CAZAC (constant amplitude zero auto correlation) sequence to complete timing estimation and frequency offset estimation, but timing synchronization performance is very susceptible to frequency offset. In [5], the training sequence constructed by CAZAC sequence has the structure of repetition in the time domain, and the pseudo-noise (PN) sequence is weighted, so that the timing function has very sharp peaks. However, the PN sequence weighting destroys the repetition of the training sequence, which leads to the low frequency offset estimation performance under the multipath channel. The authors [6] use two different CAZAC sequences for timing synchronization, also has quite sharp peaks, but the weighted operation of CAZAC sequences is too complex to achieve on the FPGA side. Tsai et al. [7] designed a combined carrier frequency offset and timing offset combined weighted least squares estimation algorithm under multipath fading channel conditions. The algorithm can achieve good performance in multipath channels, but with high complexity. Lv et al. [8] proposed a coarse synchronization and fine synchronization multi-level estimation algorithm, where coarse synchronization uses LS algorithm to quickly obtain coarse synchronization and coarse frequency offset estimation, and then use the ML algorithm to obtain the final fine synchronization results. Simulation results show that the algorithm is more suitable for achieving higher accuracy synchronization in a fast time varying multi-path channel environment. Schmidl and Cox [9] use the correlation of repeated CAZAC sequences as a measure function, and the implementation of FPGA is relatively simple, but its synchronization accuracy is low.

Considering timing synchronization algorithm in [9] and the data block structure recommended by IEEE802.16 [10], this paper designs a timing synchronization algorithm and implements the algorithm on FPGA platform. In the implementation process, this paper uses the sliding window accumulation, quantization and amplitude reduction techniques to reduce the computational complexity and hardware overhead of the timing synchronization algorithm. The experimental results verify the availability of [11] synchronization algorithm in the actual hardware environment.

2. SC-FDE System Overview

In this paper, the 3-channel model SUI-3 recommended by IEEE802.16 is used in the simulation [12],

and the maximum multipath delay is extended to 0.9 μs, which is within the guard interval duration N_g. The specific parameters are shown in Table 1.

Table 1. SC-FDE system parameters

Parameter	Specification
Channel bandwidth (MHz)	2.5
Sampling interval (μs)	0.4
Mapping method	16 QAM
Data rate (Mbps)	10
Data block length, N_{total}	256
FFT size, N_{FFT}	256
UW length, N_g (T_g)	32
Frame length	1,000 symbols
Multipath delay, t (μs)	[0 0.4 0.9]
Omnidirectional antenna power, P (dB)	[0 -5 -10]
Doppler frequency shift, fm (Hz)	[0.4 0.3 0.5]

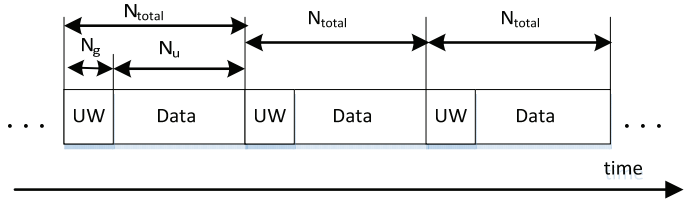

Fig. 1. SC-FDE frame structure.

Fig. 1 is the frame structure of SC-FDE. Each data block adds the same unique word (UW) suffix to form a symbol [13,14]. The UW suffix that is added to each data block is the same, so each symbol has ideal periodicity. When the maximum delay N_h of the channel is within the UW length range of N_g, that is $N_h < N_g$, the interference of the previous symbol data to the current data block can be eliminated. The UW sequence consists of CAZAC sequences, which can be obtained by Eq. (1). CAZAC sequences has good correlation and broadband and stationary frequency response, so it can be used as UW suffix, frequency offset tracking and frequency domain equalization [15].

$$c_n = e^{j\pi n^2/N}; n = 0,1,\cdots,N-1. \tag{1}$$

SC-FDE system block diagram shown in Fig. 2. In the transmitter path, the binary input data is firstly encoded by convolution. After coding, the binary values are converted into QAM (quadrature amplitude modulation) values, and a guard period is added between successive blocks. After pulse shaping (root-raised-cosine pulses) and digital-to-analog conversion, the resulting I/Q signals are up-converted to RF. In the receiver path, after passing the RF part and the analog-to-digital conversion, the receiver uses a special frame structure for time frequency synchronization and channel estimation. The received data that removes the prefix is then transformed into the frequency domain, and the channel state information (CSI) obtained by channel estimation is used for frequency domain equalization. Finally, the received data is converted to the time domain, and the QAM mapping and Viterbi decoding are performed [16].

Assuming that the transmitted data is x_n and the channel impulse response is h_n, when the number of symbols is less than the length of the impulse response, each received data symbol may be expressed as:

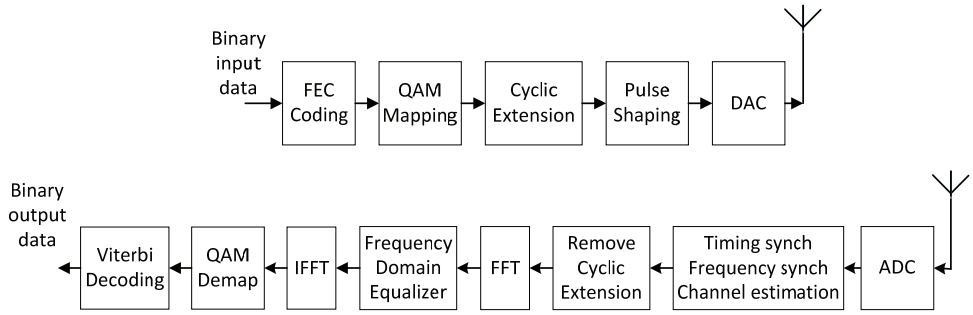

Fig. 2. SC-FDE system diagram.

$$y_n = h_n \otimes x_n + v_n; n = 0,1,\cdots,N-1. \tag{2}$$

where v_n is the additive Gaussian white noise, $\otimes$ is the convolution symbol. After FFT transformation, the frequency domain receive signal Y_k is expressed as:

$$Y_k = X_k H_k + V_k; k = 0,1,\cdots,N-1. \tag{3}$$

where H_k is the channel frequency domain response and V_k is the noise frequency domain response. It is assumed that both time frequency synchronization and channel estimation are ideal, the equilibrium structure of the judgment feedback is used, the signal before the decision is:

$$Z_n = \frac{1}{N}\sum_{k=0}^{N-1} W_k Y_k e^{j\frac{2\pi}{N}kn} - \sum_{k\in F_B} c_k \hat{x}_{n-k} \tag{4}$$

where W_k is the feedforward frequency domain filter and c_k is the coefficient of the feedback time domain filter [17].

3. Timing Synchronization Scheme

3.1 Preamble Sequence Structure for Timing Synchronization

As shown in Fig. 3, the preamble sequence contains short preamble and long preamble. The short preamble consists of 8 repetitive short-training symbols A, each of which has 32 sampling points. The entire short preamble has 256 sampling points, which can complete the automatic gain control (AGC), coarse synchronization, coarse frequency offset estimation and other functions [18]. The long preamble consists of four repetitive long-training symbols C, each of which has 64 sampling points. The entire long preamble has 256 sampling points and can be used for fine synchronization, fine frequency offset estimation and channel estimation. Both the short and long preamble are composed of CAZAC sequences.

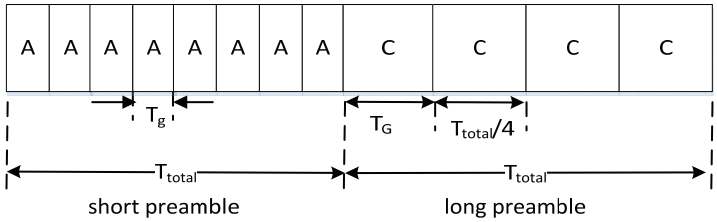

Fig. 3. Preamble sequence of SC-FDE.

3.2 Coarse Synchronization Algorithm

The coarse synchronization can be realized by the time delay autocorrelation algorithm, and the block diagram of the algorithm is shown in Fig. 4. The window C is the correlation coefficient between the received signal and the received signal after D time delay. The delay z^{D} is equal to the period of the short preamble, where D = 32. The window P calculates the energy of the received signal during the correlation coefficient window. The resulting energy values are used for normalizing the decision statistics, so that the decision variable m_n is independent of the received power.

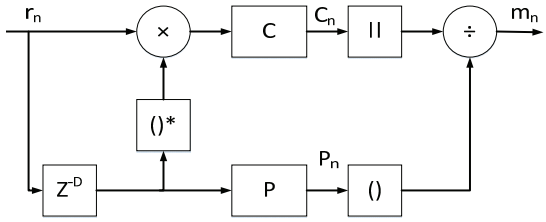

Fig. 4. Block diagram of delay autocorrelation algorithm.

The value of delay correlation C_n is as follows:

$$C_n = \sum_{k=0}^{L-1} r_{n-k} r_{n-k-D}^{*} \tag{5}$$

Eq. (5) is the cross-correlation between the currently received L data, and the L data received before D times.

The value of the received signal energy P_n can be expressed as:

$$P_n = \sum_{k=0}^{L-1} r_{n-k-D} r_{n-k-D}^{*} = \sum_{k=0}^{L-1} |r_{n-k-D}|^2 \tag{6}$$

The decision variable m_n of the delay correlation algorithm is:

$$m_n = \frac{|C_n|}{P_n} = \frac{\left|\sum_{k=0}^{L-1} r_{n-k} r_{n-k-D}^{*}\right|}{\sum_{k=0}^{L-1} r_{n-k-D} r_{n-k-D}^{*}} \tag{7}$$

At low SNR conditions, the decision variable m_n may be affected by a larger random noise in the channel and exceeds the preset threshold, so that the receiver cannot correctly judge the arrival of data. Therefore, in order to reduce the false alarm probability and improve the reliability of the coarse synchronization algorithm under the condition of low SNR, it can increase the requirement of keeping

the length on the basis of the delay correlation algorithm. As shown in Fig. 5, the decision variable m_n is required to maintain a certain number of sampling cycles above the preset threshold to determine that coarse synchronization is completed, thus avoiding the effect of large random noise.

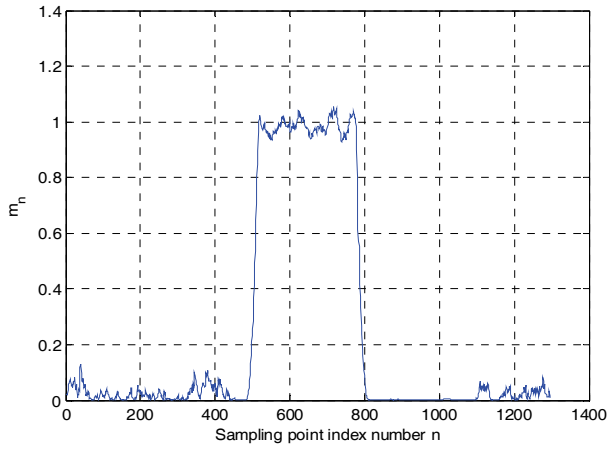

Fig. 5. Simulation diagram of coarse synchronization algorithm.

3.3 Fine Synchronization Algorithm

The coarse synchronization algorithm based on delay correlation and length retention can only provide a rough estimate of the start of the received data, while the exact timing of the received data is done by fine synchronization. The fine synchronization algorithm is implemented by the cross-correlation between the received data and the local sequence. The correlation coefficient is obtained as follows:

$$C_k = \sum_{m=0}^{M-1} r_{k-m} \times S_m^* \tag{8}$$

where superscript * indicates conjugation, M is the length of the correlation coefficient, and the size of the M determines the performance of the fine synchronization algorithm. The greater the value of M, the better the synchronization performance, the greater the amount of calculation. M = 64 is the length of the long training symbol.

The peak value of $|C_k|$ represents the end point of a long training symbol, with this feature, you can find the end point of all long training symbols in the long preamble. The last peak time of the $|C_k|$ is the end point of the long preamble.

Before timing synchronization, the received signal is not corrected for frequency offset, because the crystal stability of the general hardware is higher. After simulation, we can know that the cross-correlation method can achieve precise synchronization under the small frequency offset. If the frequency offset is very large, the frequency offset estimation and compensation can be made by short training after coarse synchronization. In the additive Gaussian white noise (AWGN) channel with a SNR of 10 dB, the MATLAB simulation results of the amplitude of the correlation coefficient between the received signal and the locally known long-training symbol are shown in Fig. 6. As can be seen from Fig. 6, the receiver includes 300 noise before the arrival of the data symbol, and 4 peaks can be obtained by cross-correlation between the long training sequence and the received data. According to the frequency and the order of the peak value, the synchronous ordinal position can be obtained.

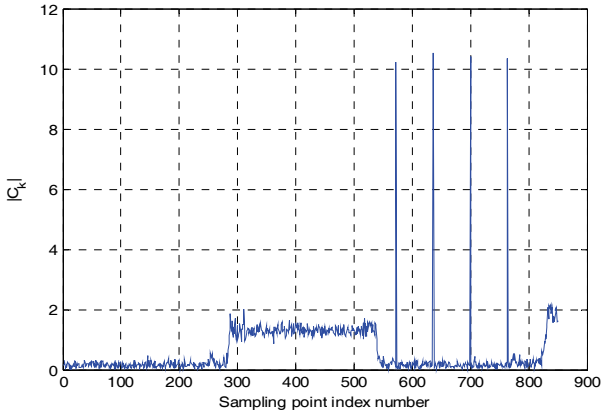

Fig. 6. Simulation diagram of fine synchronization algorithm.

4. Timing Synchronization of FPGA Design

4.1 FPGA Design of Coarse Synchronization Algorithm

As shown in Eq. (9), coarse synchronization decision variable m_n is not only larger than the threshold T_h, but also must keep a certain length of time. Referring to the experience values of multiple simulations, the retention time value T of this article is 50.

$$m_n = \frac{|C_n|}{P_n} > T_h \tag{9}$$

A large number of complex multipliers are used in the synchronization process. As shown in Eq. (10), the implementation of the usual complex multiplication requires 4 multipliers. In order to reduce the use of multipliers, we convert (10) into Eqs. (11) and (12). As shown in Fig. 7, only three multipliers and some additional addition and subtraction operations can achieve the multiplication of complex data.

$$Z_r + Z_i j = (A_r + A_i j) \times (B_r + B_i j) \tag{10}$$

$$Z_r = A_r B_r - A_i B_i = A_r(B_r + B_i) - B_i(A_r + A_i) \tag{11}$$

$$Z_i = A_r B_i - A_i B_r = A_r(B_r + B_i) - B_r(A_r - A_i) \tag{12}$$

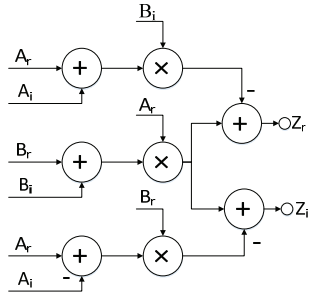

Fig. 7. Complex multiplication structure.

Correlation value accumulation calculation mainly completes the accumulation calculation of the correlation value within the length of the delay window. In order to reduce hardware overhead, idea of sliding window accumulation is considered in hardware implementation. Sliding window structure shown in Fig. 8.

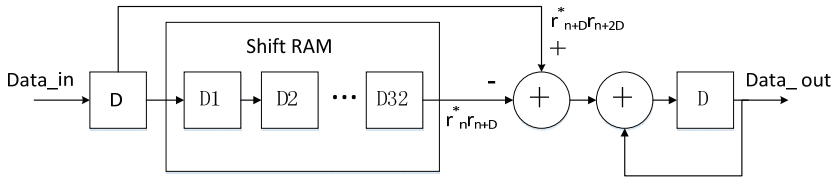

Fig. 8. Structure of sliding window accumulation algorithm.

First, the correlation values are sent to a shift register of depth 32, and the data delay is achieved by shift registers. Then send the current correlation value and the delay 32 level correlation value to the accumulation window to achieve accumulation. The sliding window accumulation equation is as follows:

$$sum = sum + (r^*_{n+D} r_{n+2D}) - (r^*_n r_{n+D}) \tag{13}$$

In Eq. (13), $r^*_{n+D} r^*_{n+2D}$ is the current correlation coefficient, and $r^*_n r^*_{n+D}$ is the correlation coefficient after the delay of stage 32.

The calculation of $|C_n|$ in Eq. (9) is not conducive to hardware implementation, so set $C_n = a_n + jb_n$ and do the following approximate:

$$|C_n| = \sqrt{a_n^2 + b_n^2} \approx |a_n| + |b_n| \tag{14}$$

After approximate simplification is completed, the estimated value $|C_n|$ is slightly larger than the actual value, so the threshold T_h must be adjusted slightly.

Directly implementing the Eq. (9) on the FPGA requires an additional divider. Since the value of T_h is predetermined, conversion of Eqs. (9)–(15) can avoid division operations. In order to save multipliers, let $T_h = 0.5$, P_n multiply by 0.5 can be achieved by right shift one bit.

$$|C_n| > P_n \times T_n = 0.5 P_n = (P_n >> 1) \tag{15}$$

where >> on behalf of the right shift operation.

Based on the above analysis, coarse synchronization hardware structure shown in Fig. 9. Hardware implementation structure of coarse synchronization algorithm mainly includes five modules: data buffer, master control, delay correlation energy calculation, correlation window energy calculation, and frame search. The data buffer module implements the caching of the input data for waiting detection. The main control module outputs the corresponding control instructions to the data buffer module according to the current state of the system and the output result of the frame search module. Delay correlation energy calculation, correlation window energy calculation and frame search three modules constitute the main body of the delay correlation algorithm, which completes coarse synchronization and feedback to the main control module.

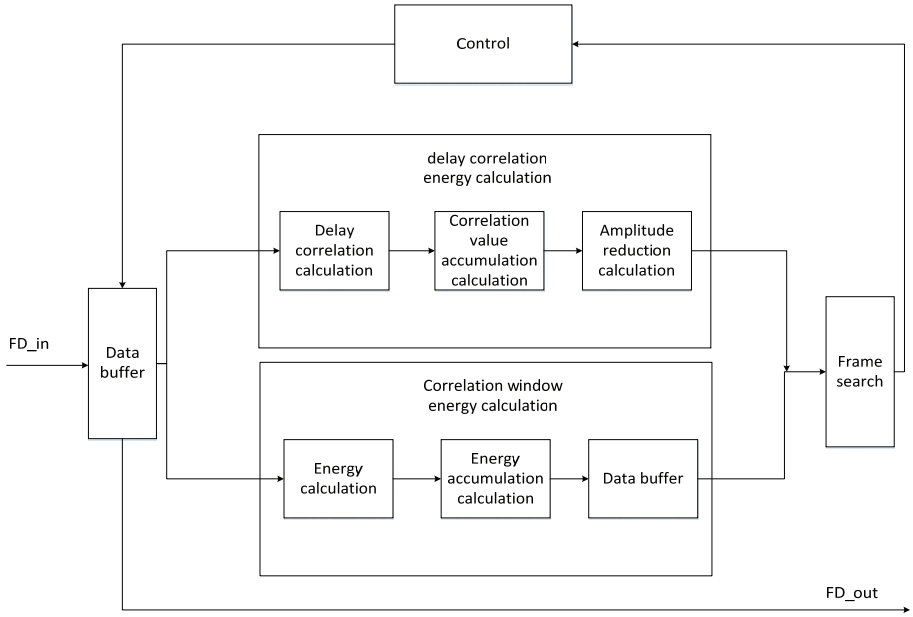

Fig. 9. Structure diagram of coarse synchronous hardware.

4.2 FPGA Design of Fine Synchronization Algorithm

When implementing algorithms for fine synchronization, there are two points to consider based on the complexity of the hardware circuit:

- Generally, the method of searching for peak value usually uses the method of searching maximum value, which requires more complicated logic circuit and control circuit for hardware implementation. Therefore, from the point of view of simplifying hardware design, this paper adopts the method of setting threshold. When the value of $|C_k|$ exceeds the preset threshold value, the peak is found.
- The implementation of the fine synchronization algorithm requires 64 complex multipliers, which take up the valuable multiplier resources on the FPGA and are also detrimental to the speed of the operation. In the hardware circuit, in order to save all of the multiplier and improve the speed of the system, the received signal is quantized as {+1, -1}.

Fig. 10 is a the MATLAB simulation results of the amplitude of the correlation coefficient between the quantized received signal and the known long training symbol in the white noise (AWGN) channel with a SNR of 10 dB. Likewise, the received signal here also includes 300 noise points before the arrival of the data symbol. After quantization, the noise is slightly higher, but there are still obvious peaks.

Based on the above analysis, Fine synchronization of the hardware structure shown in Fig. 11. The hardware implementation of the fine synchronization algorithm can be divided into three parts: quantization, matching filtering, and symbol output.

Correlation value accumulation calculation are the first part of the matched filtering module, which mainly realizes the correlation value accumulation of the received data with the local short training symbol. First, the quantized received signal is shifted into the shift register and then multiplied by the 64 Local training symbols. In the specific hardware implementation, the received data after quantization is

limited to four cases: $1 + j$, $1 - j$, $-1 + j$ and $-1 - j$. So the multiplication can be simply implemented by the addition operation, thus saving the hardware overhead. If the local long training symbol is sampled as $a + jb$, the cross-correlation operation corresponding to the four cases is:

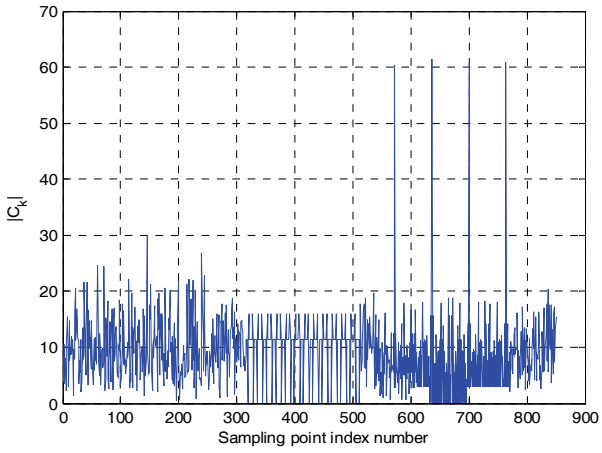

Fig. 10. Using the quantitative processing of fine synchronization algorithm simulation diagram.

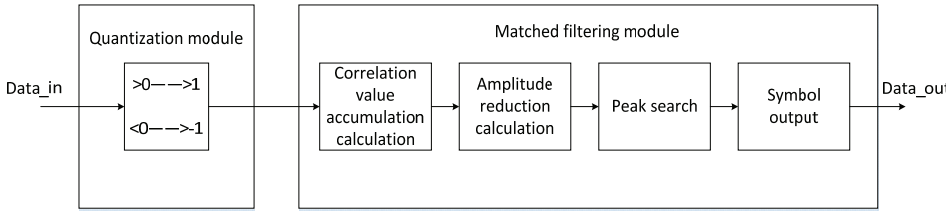

Fig. 11. Fine synchronization hardware implementation diagram.

$$(a + jb)^* \times (1 + j) = (a + b) + j(a - b) \tag{16}$$

$$(a + jb)^* \times (1 - j) = (a - b) + j(-a - b) \tag{17}$$

$$(a + jb)^* \times (-1 + j) = (-a + b) + j(a + b) \tag{18}$$

$$(a + jb)^* \times (-1 - j) = (-a - b) + j(-a + b) \tag{19}$$

As can be seen from Fig. 12, the method of quantizing the received signal will have a slight effect on the timing synchronization performance at low SNR. After 2,000 independent experiments, the simulation results show that the timing accuracy can reach more than 85% in the AWGN multipath channels of SNR = 3. When the SNR = 8 or more, in AWGN multipath channels, timing accuracy can reach almost 100%. So, the impact of quantization on synchronization performance is negligible.

5. Simulation Results

On the basis of the above analysis and discussion, the program is written on Quartus-II development software, and then the timing synchronization algorithm is simulated on ModelSim software.

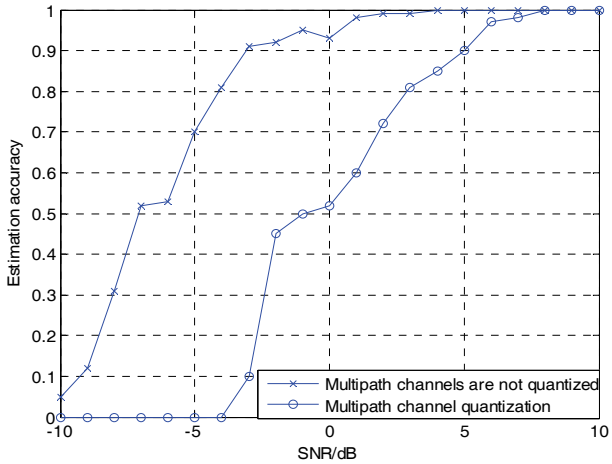

Fig. 12. Comparison of synchronization performance before and after quantization.

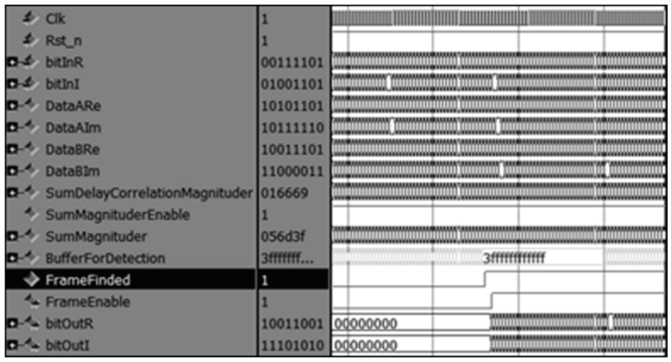

Fig. 13. ModelSim simulation result of coarse synchronization module.

The coarse synchronization simulation diagram is shown in Fig. 13. DataARe and DataAIm are the real part and the imaginary part of the current data, DataBRe and DataBIm are the real and imaginary parts of the data after the delay of level 32, SumDelayCorrelationMagnituder is the delay correlation energy value $|C_n|$ and SumMagnituder is the correlation window energy value P_n. Eq. (9) shows that m_n is a decision variable, and T_h is a threshold. BufferDetection is used to calculate the continuous points of m_n greater than threshold T_h by means of the shift method. When BufferDetection = 50'h3fffffffffff, that is, the decision variable m_n keeps the number of 50 sampling periods above the preset threshold value 0.5, satisfying the condition of keeping length in Fig. 5. At this point, the coarse synchronization is judged to be completed. The fine synchronization simulation is shown in Fig. 14. Where DataInRe and DataInIm represent the real and imaginary parts of the input data, QuantizationRe and QuantizationIm are the real and imaginary parts of the quantized data, CorrelationSumRe and CorrelationSumIm are cumulative value of correlation calculation for the quantified input data and local 64 long-training symbol sampling, STS_end_counter is the number of detected peaks. The received data after quantization is correlated with the local 64 long-training symbols sampling points. When STS_end_counter = 4, these 4 peaks correspond to the four peaks in the Fig. 10. At this point, the fine synchronization is judged to be completed. After that, the long training symbols and data symbols are serially output. The hardware

design is based on Altera's Cyclone V series 5CGXFC5C6F27C7N chip, occupying resources as shown in Table 2.

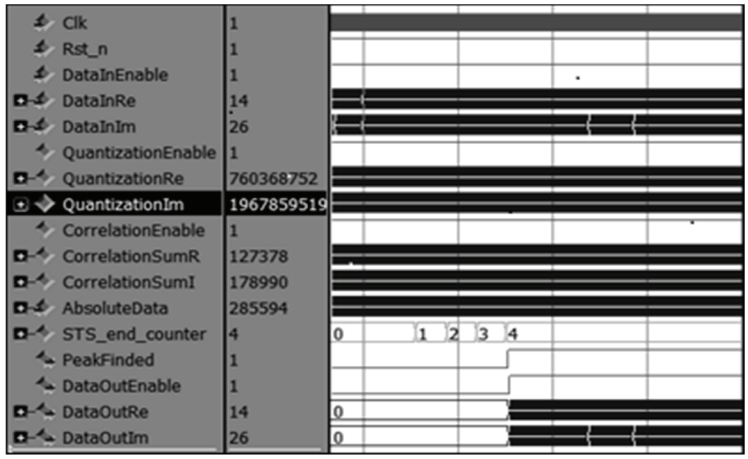

Fig. 14. ModelSim simulation results of the fine synchronization module.

Table 2. Algorithm major resource occupancy

Resource	Resource occupation
Logicutilization (inALMs)	1027/29080 (3%)
Registers	1272
Memorybits	3262/4567040 (<1%)
DSPBlock	4/150 (3%)

From Table 2 we can see that the timing algorithm occupies DSPBlock 4, occupancy rate of 3%. This shows that the algorithm can save a lot of hardware resources and is conducive to practical engineering applications.

6. Conclusion

In this paper, the coarse synchronization algorithm and the fine synchronization algorithm of SC-FDE timing synchronization are analyzed, and the FPGA design scheme of timing synchronization algorithm is given. In order to save the hardware resources and improve the speed of operation, the sliding window accumulation, quantization processing and amplitude simplification are used in the hardware implementation. As can be seen from the simulation Fig. 12, the timing accuracy can reach more than 85% in the AWGN multipath channels of SNR = 3. When the SNR = 8 or more, in AWGN multipath channels, timing accuracy can reach almost 100%. The simulation results verify that the synchronization algorithm has good performance in the actual hardware environment. The timing synchronization algorithm implemented on FPGA in this paper is more suitable for applications in high SNR scenes. The future research direction is to improved synchronization algorithm and implement the synchronization algorithm with higher timing accuracy under low SNR on FPGA hardware platform.

Acknowledgement

This paper is supported by National Key Technology Research and Development Program of the Ministry of Science and Technology of China (No. 2015BAK05B01).

References

[1] J. Coon, J. Siew, M. Beach, A. Nix, S. Armour, and J. McGeehan, "A comparison of MIMO-OFDM and MIMO-SCFDE in WLAN environments," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM'03) (IEEE Cat. No. 03CH37489)*, San Francisco, CA, 2003, pp. 3296-3301.

[2] J. J. Van De Beek, M. Sandell, M. Isaksson, and P. O. Borjesson, "Low-complex frame synchronization in OFDM systems," in *Proceedings of the 4th IEEE International Conference on Universal Personal Communications*, Tokyo, Japan, 1995, pp. 982-986.

[3] Y. Guo, G. Liu, and J. Ge, "A novel time and frequency synchronization scheme for OFDM systems," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 2, pp. 321-325, 2008.

[4] J. Meng and G. Kang, "A novel OFDM synchronization algorithm based on CAZAC sequence," in *Proceedings of 2010 International Conference on Computer Application and System Modeling*, Taiyuan, China, 2010, pp. 634-637.

[5] G. Ren, Y. Chang, H. Zhang, and H. Zhang, "Synchronization method based on a new constant envelop preamble for OFDM systems," *IEEE Transactions on Broadcasting*, vol. 51, no. 1, pp. 139-143, 2005.

[6] Y. Zhu, H. Zhang, Y. Luo, "An OFDM timing and frequency synchronization algorithm based on CAZAC sequence," *Computer Simulation*, vol. 26, no. 11, pp. 130-133, 2009.

[7] P. Y. Tsai, H. Y. Kang, and T. D. Chiueh, "Joint weighted least-squares estimation of carrier-frequency offset and timing offset for OFDM systems over multipath fading channels," *IEEE Transactions on Vehicular Technology*, vol. 54, no. 1, pp. 211-223, 2005.

[8] T. Lv, H. Li, and J. Chen, "Joint estimation of symbol timing and carrier frequency offset of OFDM signals over fast time-varying multipath channels," *IEEE Transactions on Signal Processing*, vol. 53, no. 12, pp. 4526-4535, 2005.

[9] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, no. 12, pp. 1613-1621, 1997.

[10] *IEEE Standard for Local and Metropolitan Area Networks - Part 16: Air Interface for Fixed Broad-Band Wireless Access Systems*, IEEE 802.16-REVd/D5 as IEEE 802.16-2004, 2004.

[11] W. Nie, H. Jin, and H. Yan, "Time synchronization algorithm for MIMO-OFDM system," *Communication Technology*, vol. 49, no. 3, pp. 374-377, 2016.

[12] M. Xia, D. Rouseff, J. A. Ritcey, X. Zou, C. Polprasert, and W. Xu, "Underwater acoustic communication in a highly refractive environment using SC–FDE," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 3, pp. 491-499, 2013.

[13] X. Liao and Y. Bai, "Improved symbol timing synchronization algorithms for SC-FDE systems," in *Proceedings of 2013 3rd International Conference on Consumer Electronics, Communications and Networks*, Xianning, China, 2013, pp. 363-366.

[14] C. Feng, J. Zhang, Y. Zhang, and M. Xia, "A novel timing synchronization method for MIMO OFDM systems," in *Proceedings of IEEE Vehicular Technology Conference*, Singapore, 2008, pp. 913-917.

[15] IEEE 802.16 Broadband Wireless Access Working Group, "Channel models for fixed wireless applications," 2003; http://www.ieee802.org/16/tga/docs/80216a-03_01.pdf.

[16] S. Yoshizawa, H. Tanimoto, and T. Saito, "SC-FDE vs OFDM: Performance comparison in shallow-sea underwater acoustic communication," in *Proceedings of 2016 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, Phuket, Thailand, 2016, pp. 1-5.

[17] C. Chen, M. Zhao, and W. Chen, "Timing synchronization for SC-FDE," *Journal of Zhejiang University (Engineering Science)*, vol. 41, no. 3, pp. 445-449, 2007.

[18] M. Huemer, H. Witschnig, and J. Hausner, "Unique word based phase tracking algorithms for SC/FDE-systems," in *Proceedings of IEEE Global Telecommunications Conference (IEEE Cat. No. 03CH37489)*, San Francisco, CA, 2003, pp. 70-74.

Suyuan Ji https://orcid.org/0000-0001-5529-4454

He received M.S. degree from Communication University of China in June 2018. He works in Shanghai Radio Equipment Research Institute. His current research interests include software defined radio, hardware design and equipment structure.

Chao Chen https://orcid.org/0000-0002-3843-3181

He received M.S. and Ph.D. degrees from Communication University of China in 2006 and 2010, respectively. He worked in the Engineering Center of Radio and Television Digital Education Ministry of Communication University of China, and got the title of associate researcher in March 2014. His current research interests include wireless communication system, wireless digital broadcasting system, audio/image processing.

Yu Zhang https://orcid.org/0000-0002-7878-2378

He received M.S. degrees from Beijing University of Technology in 2011. He worked in the Communication and Technology Bureau of Xinhua News Agency. His current research interests include design and application of wireless multimedia communication technology, streaming media video coding and transmission technology in the news and media industry.