International Journal of Advanced Science and Technology
Vol.88 (2016), pp.1-12
http://dx.doi.org/10.14257/ijast.2016.88.01

IP Design and Implementation of a LTE-A Cell Blind Detect Scheme

Wenqiang Dai[*] and Mingbo Gou

College of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
**man199084@163.com*

Abstract

The current LTE-Advanced system architecture tends to flatten and the data transfer rate of mobile communication system continues to increase, which needs to complete cell blind detect more accurate and quick, and indicates that the design of the LTE-A system terminal need to be updated. Therefore, this paper will find a new cell blind detect scheme, then designing a corresponding IP which considers the hardware performance, area, power and scalability from the perspective of ASIC implementation, and using ASIC tools to verification and logic synthesis. Implementation results show that the designed IP can be used for a mobile terminal chip design.

Keywords: *LTE-A; Frame synchronization; Timing synchronization; ASIC*

1. Introduction

In the mobile communication system, cell search is the premise for establishing a communications link between UE and a base station, and the performance of cell search is determined by the quality of cell blind detect. Only by completing the cell blind detect accurately and quickly, UE can obtain more detailed information and neighboring cell information of the cell, and listen for paging or initiate a call.

The current LTE-Advanced system architecture tends to flatten and the data transfer rate of mobile communication system continues to increase, which indicates the previous cell blind detect implementations scheme of the LTE-A system terminal need to be updated. Therefore, it is very important to find a new cell blind detect implementation scheme.

In the LTE-Advanced system, a cell blind detect includes timing synchronization, frame synchronization, and the cell ID detection. Due to their good autocorrelation, we can obtain time synchronization, frame synchronization and the cell ID by detecting the primary synchronization signal (PSS) and the secondary synchronization signal (SSS) independently. For PSS, reference [10] proposes a calculation algorithms and architectures of PSS; reference [11] gives a lower computational complexity algorithm, which based on the mirror symmetry of PSS to obtain the PSS position, then cross-correlated with the local PSS to get the sub cell ID. For SSS, reference [5] describes two detection algorithms: coherent and non-coherent detection algorithms, which uses descrambling method. In addition, it proposes an effectively algorithm which can reduce complexity, which bases on these two detection algorithms.

First of all, due to timing synchronization needs a lot of FFT and related calculations, this paper divides the algorithm into three steps: (1) coarse timing synchronization and detecting the sub cell ID; (2) fine timing synchronization; (3) frame synchronization and

[*] Corresponding Author

detecting the cell ID group. This above-divided we adopt can reduce the amount of fine timing synchronization calculation and accelerate to complete the cell blind detect considerably. Finally, this paper will design a corresponding IP which considers the hardware performance, area, power and scalability from the perspective of ASIC implementation, and use ASIC tools to verification and logic synthesis.

2. Related Algorithms

Physical Layer Protocol of LTE-Advanced system provisions, which has 504 physical layer cell IDs. Each physical layer cell ID (used N_{ID}^{cell} to represent) could be formed by a physical layer cell ID group $N_{ID}^{(1)}$ and a physical layer sub cell ID $N_{ID}^{(2)}$ [1]:

$$N_{ID}^{cell} = 3N_{ID}^{(1)} + N_{ID}^{(2)} \tag{1}$$

In the LTE-Advanced system, information associated with the cell ID is included in the PSS and the SSS. The PSS and SSS have a fixed position in the time-frequency domain. PSS which generation period is 5ms is decided by root index $N_{ID}^{(2)}$; SSS which generation period is 10ms is decided by the root index $N_{ID}^{(1)}$ and $N_{ID}^{(2)}$.

Before the calculation, we need to process the received data with down-sampling rate 1/16, which still meets the Nyquist sampling theorem, and does not occur mixing. While setting the sliding step is 16 Ts.

2.1. Coarse Timing Synchronization and sub cell ID Detection

It is quickly that getting the approximate location of the PSS and obtaining the sub cell ID $N_{ID}^{(2)}$ by using coarse timing synchronization, so as to determine the sliding range of the fine timing synchronization. Therefore, this paper adopts a coarse timing synchronization scheme which bases on the symmetry-related of received PSS to reduce the amount of calculation [2].

Specific steps of the scheme are: receiving half frame data (assuming that the half-frame data contained a complete PSS), with the first point of data as a starting point, followed by removing 2048 point, with r(n) representation and divided r(n) two portions. Then doing a sliding correlation calculation for the two parts of the data, the maximum value of calculation result is the approximate location of the PSS, the correlation function by the following formula (2):

$$C(d) = \left| \sum_{n=0}^{N-1} r(d+n) r^{*}(2047 - n + d) \right| \tag{2}$$

Which N represents the correlation window length, *i.e.*, is the length of half OFDM symbol. When the down-sampling rate is set to 1/16, take 64. The position of PSS approximate can obtained from the formula (3):

$$\hat{d} = \arg\max_{d}\{C(n)\} \tag{3}$$

Simulating the coarse timing synchronization scheme with MATLAB, simulation condition setting: channel is a Gaussian white noise channel, SNR is -10dB, CP for general CP, timing offset is set to 0, offset is set to 2000Hz, the $N_{ID}^{(2)}$ used for PSS of send signal is 1. The simulation is shown in Figure 1:

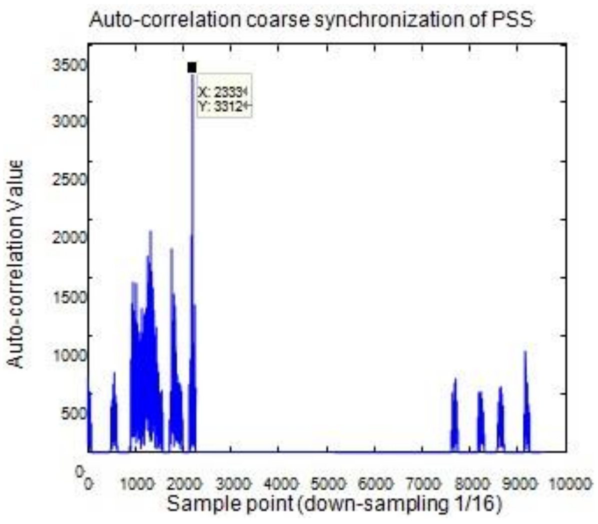


Figure 1. Simulation of Coarse Timing Synchronization Algorithm based on the Symmetry-Related of Received PSS

As we can observe in the Figure 1, the maximum value is very clear, the maximum value of the abscissa is 2333, which the position in the received data by converting is 35280. The actual location of the PSS is 35265, which has a 15 points difference with simulation results. As a result of down-sampling processing, which sampling rate is 1/16, 15-point difference can be accepted.

After finding out the approximate location of the PSS, the group the maximum value located corresponding to u, which is the root index of the received PSS, and the value of $N_{ID}^{(2)}$ can be obtained by correspondence between u and root index.

2.2. Fine Timing Synchronization

Coarse timing synchronization just has found the approximate location of the PSS, but synchronization accuracy does not meet the requirements; it also needs fine timing synchronization. In this case, the fine timing synchronization can be reduced to $[\hat{d}*16-64,\ \hat{d}*16+63]$, $\hat{d}$ represents the point of coarse timing synchronization. This paper adopts related algorithm [3] between the received PSS and the local PSS to fine timing synchronization.

First, we use the $N_{ID}^{(2)}$ obtained from coarse timing synchronization to generate locale frequency-domain PSS, and then convert it to the time domain by IFFT. Then, making a sliding correlation calculation between the received data before down-sampling and the 128 point time-domain PSS in the range $[\hat{d}*16-64,\ \hat{d}*16+63]$. Correlation function shown by the formula (4):

$$C(d) = \sum_{n=\hat{d}*16-64}^{\hat{d}*16+63} \left| r(d+n)s^{*}(n) \right|^{2} \tag{4}$$

The maximum value calculated by the formula (5) is the position of fine synchronization:

$$\hat{d}_{PSS} = \arg\max_{d}\{C(d)\} \tag{5}$$

Getting the exact location of the PSS indicates that we have completed the half-frame synchronization. Simulating the coarse timing synchronization scheme with MATLAB, simulation condition setting: channel is a Gaussian white noise channel, SNR is -10dB, CP for general CP, timing offset is set to 0, offset is set to 2000Hz, the $N_{ID}^{(2)}$ used for PSS of send signal is 1. The simulation is shown in Figure 2:

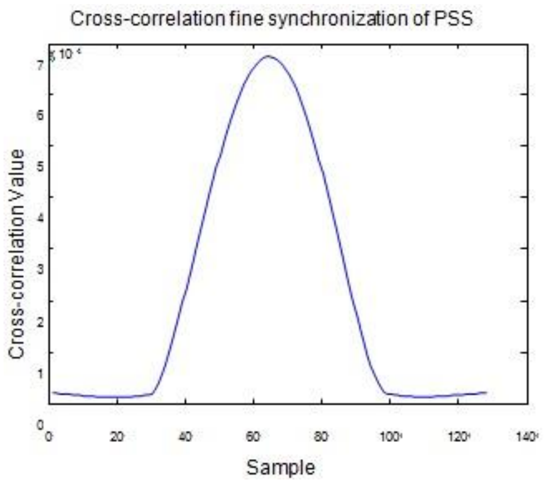


Figure 2. Fine Timing Synchronization Simulations

2.3. Frame Synchronization and Cell Group ID Detection

After timing synchronization, we have found out the exact position of the PSS, but cannot determine the current received data is the first half frame or the second half frame of the frame data. So we need to complete frame synchronization, which can obtain the $N_{ID}^{(1)}$ by detecting the SSS. Traditional SSS detection algorithm is that making correlation calculation between the received SSS and local SSS: according to the position of the maximum value, to accomplish frame synchronization. This method is computationally intensive, in order to reduce complexity and the amount of calculation, this paper decided to adopt the unscrambling way to get the parameters m_0 and m_1 of SSS generation formula, according to one relationship m_0 and m_1 with $N_{ID}^{(1)}$, to get the cell ID group $N_{ID}^{(1)}$.

Descrambling detection algorithm includes coherent detection algorithm and no-coherent detection algorithm. Reference [5] shows that, the two algorithms are that MSE decreases with increasing SNR, but coherent detection algorithm performance is better than the performance of non-coherent detection algorithm in the Gaussian channel environment. Therefore, we adopt coherent detection algorithm [4].

Coherent detection algorithm works as follows: when the coherence time of the channel is greater than four OFDM symbols, we can use PSS to obtain estimate value $\hat{H}_{PSS}(k)$ of the channel impulse response, to frame synchronization by the coherent detection method [4].

Transforming the received PSS of time-domain to the frequency-domain by FFT, which indicated by $R_{PSS}(k)$ and generating local frequency-domain PSS, which indicated by $T_{PSS}(k)$. When the coherence time of the channel is greater than the length of 4 OFDM symbols, we can calculate the estimates value of channel impulse response by formula (6):

$$\hat{H}_{PSS}(k) = R_{PSS}(k) / T_{PSS}(k) \qquad (6)$$

According to CP type and the position of PSS, finding out the time-domain SSS and transferring it to the frequency-domain SSS by FFT, making compensation to channel:

$$\hat{R}_{SSS}(k) = R_{SSS}(k)\hat{H}_{PSS}(k) \tag{7}$$

Divided $\hat{R}_{SSS}(k)$ into $\hat{R}_{SSS}(2k)$ which constituted by even bit sequence and $\hat{R}_{SSS}(2k+1)$ which constituted by odd bit sequence. According to the $N_{ID}^{(2)}$ which obtained from the timing synchronization, to generate scrambling sequence $c_0(k)$ and to descramble the $\hat{R}_{SSS}(2k)$:

$$a_{m0}(k) = \hat{R}_{SSS}(2k)c_0(k) \tag{8}$$

Making correlation calculation between $a_{m0}(k)$ and different cyclic shift sequence $S^{(i)}(k)$ of sequence m, which can obtain the estimated value $\hat{m}_0$ of m_0 according to the maximum value:

$$\hat{m}_0 = \arg\max_i\{\sum_{j=0}^{M-1} |\sum_{k=jN_M}^{(j+1)N_M} a_{m_0}(k)S^{(i)}(k)|^2\} \tag{9}$$

Which, $i = 0,1,\dots,30$, M represents the number of relevant segments, N_M represents the length of each segment of data, assumptions $M = 4$ here.

Generating local scrambling sequence $c_1(k)$ and $z_1^{(\hat{m}_0)}(k)$, and descramble $\hat{R}_{SSS}(2k+1)$:

$$a_{m1}(k) = \hat{R}_{SSS}(2k+1)c_1(k)z_1^{(\hat{m}_0)}(k) \tag{10}$$

Making correlation calculation between $a_{m1}(k)$ and different cyclic shift sequence $S^{(i)}(k)$ of m sequence, which can obtain the maximum value $\hat{m}_1$ of m_1 according to the estimated value:

$$\hat{m}_1 = \arg\max_i\{\sum_{j=0}^{M-1} |\sum_{k=jN_M}^{(j+1)N_M} a_{m1}(k)S^{(i)}(k)|^2\} \tag{11}$$

According to reference [1] tables 6.11.2.1-1, the correspondence between m_1 and the estimated value $\hat{m}_0$ of m_0 is as follows：

$$\begin{cases} m_1 \in [0,1,\cdots,\hat{m}_0+7], & \text{when } \hat{m}_0 \in [0,\cdots,2], \text{and } m_1 \neq \hat{m}_0 \\ m_1 \in [0,1,\cdots,\hat{m}_0+6], & \text{when } \hat{m}_0 \in [3,\cdots,7], \text{and } m_1 \neq \hat{m}_0 \\ m_1 \in [\hat{m}_0-7,\cdots,\hat{m}_0+6], & \text{when } \hat{m}_0 \in [8,9], \text{and } m_1 \neq \hat{m}_0 \\ m_1 \in [\hat{m}_0-6,\cdots,\hat{m}_0+6], & \text{when } \hat{m}_0 \in [10,\cdots 24], \text{and } m_1 \neq \hat{m}_0 \\ m_1 \in [\hat{m}_0-6,\cdots,30], & \text{when } \hat{m}_0 \in [25,\cdots 30], \text{and } m_1 \neq \hat{m}_0 \end{cases} \tag{12}$$

From the above equation, when $\hat{m}_0$ is less than $\hat{m}_1$, the received SSS located in sub-frame 0, otherwise located in sub-frame 5.

As $M = \hat{m}_0 - \hat{m}_1$, then $N_{ID}^{(1)}$ can be obtained from the following equation:

$$N_{ID}^{(1)} = 30(M-1) + \hat{m}_0 - (M-1)(M-2)/2 \tag{13}$$

It needs 44-related operations (estimating m_0 which needs 30 times related operations; estimating m_1 which needs 14 times related operations) by using coherent detection algorithm descrambling, and the calculation account was relatively low.

2.4. Algorithm Summary

In summary, the algorithm implementation of specific process is: first, using algorithms which bases on the relevant PSS symmetry to complete coarse timing synchronization; second, according to the algorithms which bases on the cross-correlation between receive PSS and local PSS to obtain the sub cell ID $N_{ID}^{(2)}$; third, generating local PSS according to the $N_{ID}^{(2)}$, using the cross-correlation between received PSS and local PSS to complete PSS fine-timing synchronization; fourthly, using the coherent detection algorithm to complete frame synchronization and obtain the cell ID group $N_{ID}^{(1)}$.

3. IP Design

To ensure the accuracy of cell blind detect module, the input and output data were chosen different accuracy for different functions of the module.

（1）The accuracy for the input and output data of FFT function is Q15, and its width is 32bit, which the high 16bit data represent the real part, and the low 16bit data represent the imaginary part.

（2）These two functions which produce local PSS sequence and local SSS sequence have no input data. The accuracy of their output data is Q15, width is 32bit, each of the real and imaginary part has 16bit.

（3）The input and output data of the function which finds the maximum value are real, which use 32bit width of data to represent.

（4）The accuracy for input data of calculating PSS sequence impulse response function is Q15 is Q15. The input and output data of the function are 32bit, and each of the real and imaginary part has 16bit.

（5）The accuracy for input data of M0 and M1 estimate function is Q15, which output data is real and width is 32bit, each of the real and imaginary part has 16bit.

The width of the data through calculation will be greater than the width of original data. To ensure a consistent between the input and output data, this module would make normalization with the maximum value of the above data.

3.1. Structure Description

Hardware architecture of cell blind detect module be showed in Figure 3, by five components: input module, interface module, control module, memory module and function module.

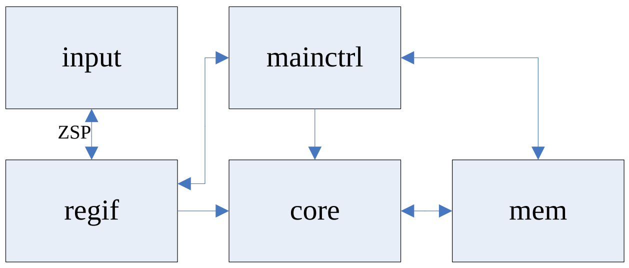


Figure 3. Cell Blind Detect Module Hardware Architecture

As we can see from the Figure 3, the interface bus of module Regif is ZSP, which can read and write data from memory or register, configure parameters, get the running status of the module by querying, and determine the end of the module operation by the interrupt flag; Module Mainctrl control the entire hardware flow and functional modules; Module Core is the core of the hardware, which completes specific functions of the hardware required to achieve; Module Mem completes ZSP bus and functional modules to access memory resources, including input and output data from memory.

3.2. Calculation Module

The calculation module is module Core, which is mainly divided into function module for FFT calculation, function module for generating a sequence of local PSS or SSS, function module for searching the maximum, function module for calculating PSS impulse response and function module for estimating the value of M0/M1.

3.2.1. FFT Function Module

The operation point includes 128, 256, 512, 1024 and 2048 that FFT function supports. This paper will use extraction times scheme based RADIX 4 algorithm [16], which bases on the traditional scheme based 2-algorithm. This allows the input transforms 2 to 4, the number accessed memory is reduced by half, thereby reducing the power consumption of the FFT operation, while in the minor performance loss case, the computation time of the FFT operation is reduced by 1/3.

After the FFT operation is completed, finding out the maximum value and the normalization factor, and outputting the result. At the same time, we use 8-parallel structure to design the module.8-parallel structure is based on parallel iterative structure, converting each level of the whole structure of parallel iterative parallel to 8-parallel, reduces the number of parallel units, which reduces chip area [12].

3.2.2. Module for Producing local PSS or SSS Sequence

According to different requirements, this module can generate a local frequency-domain synchronous sequence or time-domain synchronous sequence. First, generating 62-point PSS or SSS sequence in the frequency domain, which depends on the configuration of the cell ID group $N_{ID}^{(1)}$ and the sub cell ID $N_{ID}^{(2)}$, as well as the formula for generating PSS sequences or SSS sequence. Then, if generating the frequency-domain sequence, complement five zero in front of and behind the 62 points frequency-domain sequence, to obtain 72 points frequency-domain sequence and output the resulting sequence; If generating the time-domain sequence, making the above 72 points frequency-domain sequence frequency-domain move, mapping to the center 72 subcarriers, and then making IFFT operation, which can obtain a 128 points of PSS or SSS time-domain signal.

3.2.3. Module for Finding the Maximum Value

The module supports 32bit real sequence which input length up to 2048 point. This function can find the main peak and two second peaks of the input data, and locate the three secondary peaks around each main peak. The function searching maximum value finds a main peak firstly, and then finds the three auxiliary peaks around the first main peak, and then finds the second secondary main peak, and so on. Each main peak interval 128 points at least.

3.2.4. Module for Calculating PSS Impulse Response Calculation

This input data of module is received time-domain PSS sequence, and its length is fixed at 128 points, the real and imaginary part of each point data has 16bit independently. At first, input data is subjected to FFT operation, to get the 128 points PSS frequency-domain sequence, and extracting the 62 point PSS sequence. Depending on the configuration of the sub cell ID $N_{ID}^{(2)}$, and the formula generating PSS sequences, to generate 62 points local PSS frequency-domain sequence. Local PSS sequence makes correction calculation with received PSS sequence, to obtain impulse response. The PSS impulse response is a 62-point data, which width is 32bit and each of the real and imaginary part has 16bit.

3.2.5. Module for Estimating M0

This input data of this module is received time-domain SSS sequence, and its length is fixed at 128 points, which the real and imaginary part of each point data has 16bit independently. At first, input data are subjected to FFT operation, to get 128 points SSS frequency-domain sequence, and extracting the SSS 62 point sequence, to get $\hat{R}_{SSS}(2k)$ on the even position. Then, according to the configuration of the sub cell ID $N_{ID}^{(2)}$ to generate scrambling sequences $c_0(k)$ for descrambling $\hat{R}_{SSS}(2k)$. We need make correlation calculation by different segment between the different cyclic shift sequences $S^{(i)}(k)$ of sequence m generating in local and descrambled sequence, outputting the correlation value of 32bit.

3.2.6. Module for Estimating M1

This module extracted 62 points SSS sequence from M0 estimation module, to get the data which located at odd position. Then, according to the sub cell ID $N_{ID}^{(2)}$ and $\acute{m}_0$ which get from estimating M0 value, generating scramble sequence $c_1(k)$ and $z_1^{(\mathring{m}_0)}(k)$, which descramble the $\hat{R}_{SSS}(2k+1)$ second time. We need make segment correlation calculation between descrambled sequence and the different cyclic shift sequences $S^{(i)}(k)$ of sequence m generating in local, and outputting the correlation value of 32bit.

3.3. Control Module

This paper uses a finite state machine (Finite Status Machine, FSM) to design control module. FSM jump of control module is shown in the Figure 4:

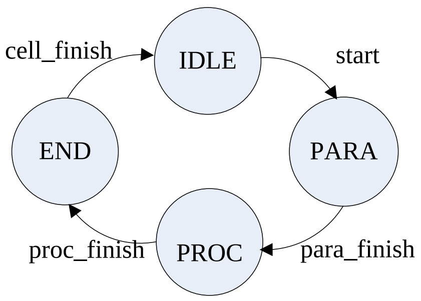


Figure 4. FSM Jump of Control Module

(1) IDLE state: Module is in idle state when the module does not start. When the start signal of the module is valid, the FSM would jump from state MEAS_PARA to state IDLE.

(2) PARA state: When entering this state, signal para_en would be set. When para_en is valid, function module would read the parameter configuration values of the corresponding function from the parameter register. After completing read parameters, signal para_finish would be set, which indicates that read the parameter has been completed, and the FSM would jump from state PARA to state PROC.

(3) PROC state: When entering this state, signal meas_en would be set. When signal meas_en is valid, the cell blind detect module would calculate particular function. After completing calculation, signal proc_finish would be set, which indicates that calculation has been completed, and the FSM would jump from state PROC to state END.

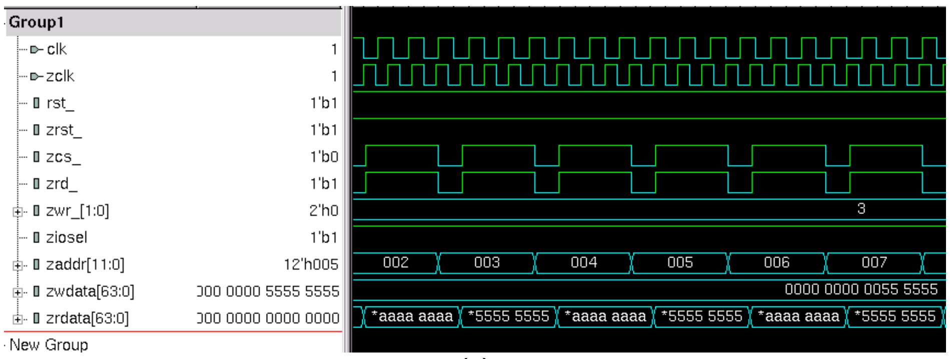
(4) END state: When entering this state, which show that the function of external configuration tasks have been completed, signal cell_finish would be set, and the FSM would jump from state END to state IDLE.

4. Experiments and Analysis

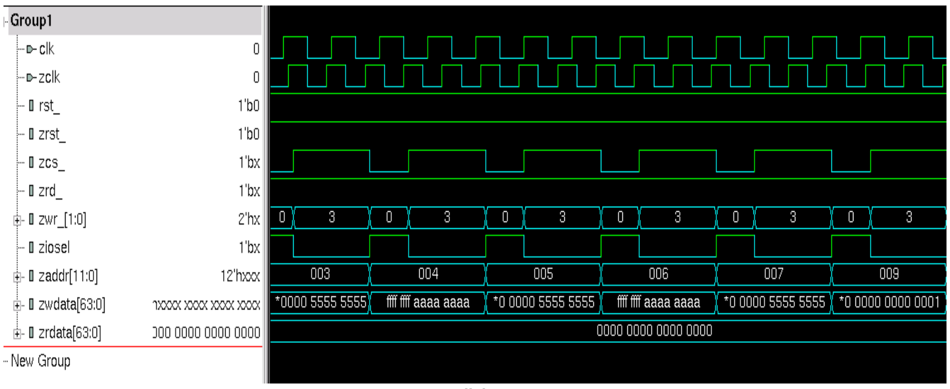
4.1. Functional Verification

This article makes a comprehensive functional verification for cell blind detect module by using VCS emulator [8, 9] and graphical debugging tool. Picking and choosing few typical waveforms for analysis and description.

4.1.1. Simulation Results of Register Basic Properties



(a)

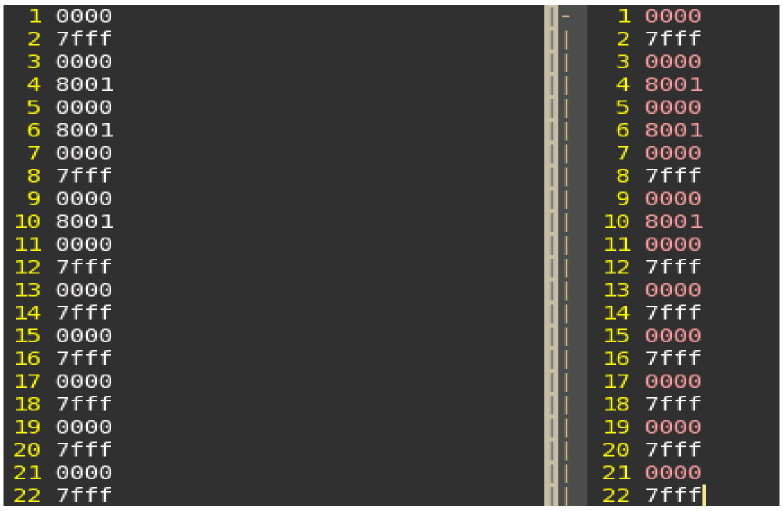


(b)

Figure 5. (a)Register Read Waveform; (b)Register Write Waveforms

As we can see from the Figure 5, the designed module meets the timing of bus interface; the registers can be set correctly and the reset value is correct; the registers can read and write properly, and read-write property can fully meet the requirements of the design.

4.1.2. Simulation Results of Producing local SSS Sequence Function



```
0000    0000
7fff    7fff
0000    0000
8001    8001
0000    0000
8001    8001
0000    0000
7fff    7fff
0000    0000
8001    8001
0000    0000
7fff    7fff
0000    0000
7fff    7fff
0000    0000
7fff    7fff
0000    0000
7fff    7fff
0000    0000
7fff    7fff
0000    0000
7fff    7fff
```

Figure 6. Output Data Comparison of SSS Series Features Local Produce

As we can be viewed in Figure 6, the data on the left side of the figure is the output data of the function, and the data on the right side of the figure is the output data of reference model. We can find that the comparison results are identical by comparing the output data of this module and the output data of the reference model data.

4.1.3. Simulation Results of M0 Value Estimate Function

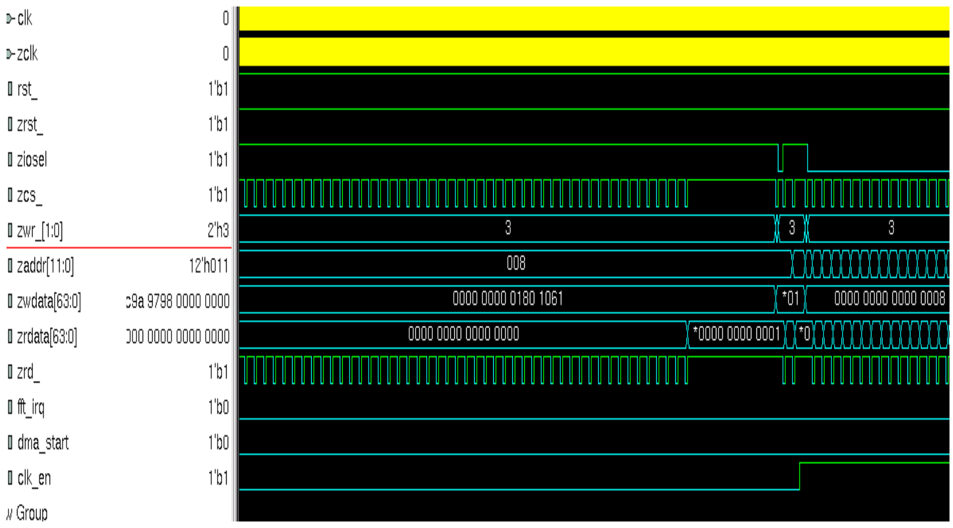


Figure 7. Simulation Waveforms M0 Value Estimate Functional

As we can be viewed in Figure 7, this simulation does not enable interrupts, so after the module runs to the end, but doesn’t generate an interrupt signal. In this case, only continue to read the value of the interrupt flag register, until the interrupt flag bit be set, which can determine the module has run to the end, and then reading the output data from the memory and compare the output data. Comparative results show that the function can achieve the desired objectives.

4.1.4. Simulation Results of M1 Value Estimate Function

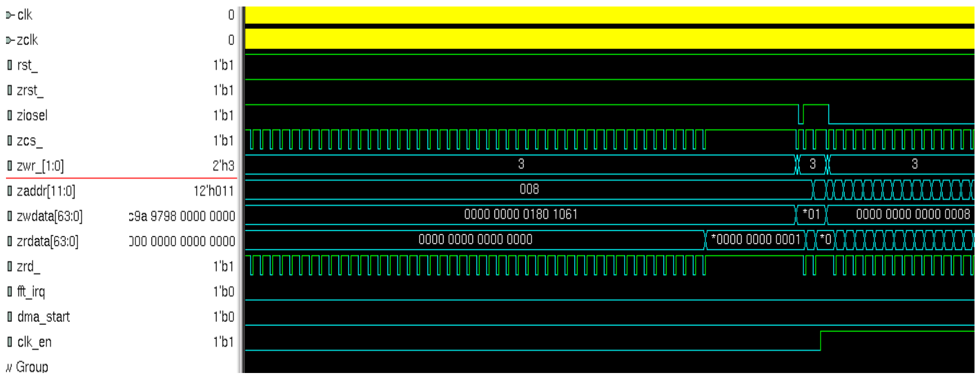
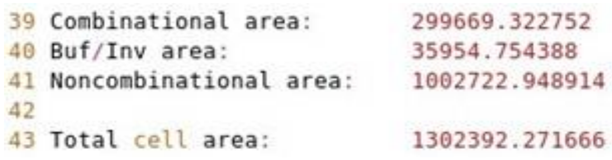


Figure 8. Simulation Waveforms M1 Value Estimate Functional

As we can be viewed in Figure 8, before running the M1 value estimates function, we need to clear the memory. After module has cleared memory, storing the input data which estimating M1 value needed in memory, and then according to the configure parameter register and the information of control register, starting the cell blind detect module. We can see from the waveform, the DMA starts the module. After the module runs to end, and generates an interrupt signal. The comparison result of the simulation shows that the function can achieve the desired objectives.

4.2. Logic Synthesis

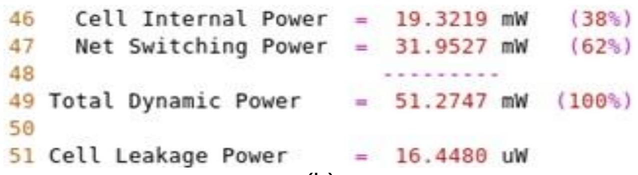
This paper uses Design Compiler synthesis tool for the cell blind detect module to logic synthesis; it can transfer RTL code into gate-level net list and generates the corresponding delay file. Logic area report and power consumption comprehensive report after logic synthesis is shown in Figure 9:



```
Combinational area:        299669.322752
Buf/Inv area:              35954.754388
Noncombinational area:     1002722.948914

Total cell area:           1302392.271666
```

(a)



```
   Cell Internal Power  =  19.3219 mW   (38%)
   Net Switching Power  =  31.9527 mW   (62%)
                          ---------
Total Dynamic Power     =  51.2747 mW  (100%)

Cell Leakage Power      =  16.4480 uW
```

(b)

Figure 9. (a)The Cell Blind Detect Module Area Report; (b) The Cell Blind Detect Module Area Report

We can draw the following conclusions from Figure 9:

(1) The logic area of this design after integrated is 1302392.271666um2, which the area of combinational logic is 299699.322752um2, the area of temporal logic is 1002722.948914um2；

(2) The dynamic power is 51.2747mw, static power is 16.448uw, and the total power is 51.2911mw.

5. Conclusions

This paper proposes a cell blind detect algorithm scheme, and designs a corresponding IP which has considered accuracy, area, and power consumption, scalability from the perspective of ASIC implementation. This paper divides the algorithm into three steps: (1) coarse timing synchronization and detecting the sub cell ID; (2) fine timing synchronization; (3) frame synchronization and detecting the cell ID group. The above-divided can find the approximate location of the PSS quickly by coarse timing synchronization, and narrow the detection range of the fine timing synchronization. The simulation results of the scheme show that the above-divided can complete timing synchronization, frame synchronization between UE and base station, and obtain the cell ID quickly and accurately. In addition, the designed IP achieves the above described functions, and its area and power consumption also have get better optimization. In summary, the data processing speed and accuracy of the design meet the testing requirements of LTE system, the designed IP can be used for designing terminal chips.

References

[1] 3GPP TS 36.211 v11.0.0. Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation (Release 11), **(2013)**, pp. 108-111.

[2] Z. Zhang, J. Liu and K. Long, “Low-complexity cell search with fast PSS identification in LTE”, Vehicular Technology, IEEE Transactions, vol. 61, no. 4, **(2012)**, pp. 1719-1729.

[3] Y. Sheng and X. Luo, “Algorithm Study on Cell Search in LTE”, Communications Technology, 3: 035.

[4] H. G. Park, I. K. Kim and Y. S. Kim, “Efficient coherent neighbor cell search for synchronous 3GPP LTE system”, Journal of Electronics Letters, vol. 44, no. 21, **(2008)**, pp. 1267-1268.

[5] C. Fa tang and M. Lei, “A new method for secondary synchronization signal detection algorithm in TD-L TE system down link”, Journal of Application of Electronic Technique, vol. 38, no. 2, **(2012)**, pp. 91-93.

[6] C. L. Chen, S. G. Chen and Y. T. Lin, “Efficient non-coherent PSS detections and analysis for LTE systems”, Signal Processing and its Applications (CSPA), 2012 IEEE 8th International Colloquium on. IEEE, **(2012)**, pp. 305-309.

[7] Synopsys. VCS User Guide Version 6.1., **(2002)**, pp. 12-18.

[8] Synopsys. VirSim User Guide Version 4.1.1., **(2002)**, pp. 5-14.

[9] J. Bergeron, “Writing test benches: functional verification of HDL models”, Dordrecht: Kluwer Academic Publishers, **(2003)**, pp. 5-20.

[10] M. M. Mansour, “Optimized architecture for computing Zadoff-Chu sequences with application to LTE”, Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE. IEEE, **(2009)**, pp. 1-6.

[11] Y. Xiumei, X. Yong and J. Guoqing, “Fast Acquisition of Primary Synchronization Signal in LTE Systems”, Journal of Applied Sciences, vol. 30, no. 1, **(2012)**, pp. 14-18.

[12] L. Jie. “The ASIC Design of Low Power and Scalable Fast Fourier Transformation”, Hunan University, **(2011)**, pp. 20-53.

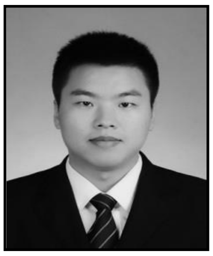
[13] M. M. Mansour, “Optimized architecture for computing Zadoff-Chu sequences with application to LTE”, Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE. IEEE, **(2009)**, pp. 1-6.

[14] Y. Xi Qing, “ASIC design practical tutorial”, Zhejiang University Press, **(2007)**, pp. 3-10.

[15] S. A. Lonkar, A. C. Uchagaonkar and K. T. V. Reddy, 2015 International Conference on Communication, Information & Computing Technology (ICCICT), Mumbai, India, January 16-17.

[16] Y. Yao and C. Fa tang, “Analysis of Turbo codes performance based on the RADIX4 algorithm”, Journal of Chongqing University of Posts and Telecommunications (Natural Science), vol. 18, no. 3, **(2006)** June.

Author



Wenqiang Dai received his B.S. degree from Changchun University of Technology, and his Ph.D. degree from Chongqing University of Posts and Telecommunications y, in 2013 and now. His main research is mobile communication terminal technology.