2010 International Conference On Computer Design And Appliations (ICCDA 2010)

Implementation of Automatic Gain Control in OFDM Digital Receiver on FPGA

Lingyan Liang, Jianghong Shi*, Lingyu Chen, Shuangyan Xu
Dept. of Communication Engineering
Xiamen University
Xiamen, China
shijh@xmu.edu.cn

***Abstract*—An appropriate design based on well-defined system parameters and architecture can make a huge difference in the performance. In particular, there is a need for improved digital Automatic Gain Control (AGC) for use in Orthogonal Frequency Division Multiplexing (OFDM) system. In this paper, we study the principle of the AGC loop, and adopt an appropriate AGC loop model which takes hardware resource and performance in considering. On this basis, we give a detail on how to design parts of AGC loop on FPGA, including implementation process and setting parameters. The performances of the locking time and gain jitter are given in theory. Then the achieving results on FPGA is shown to improve the design is feasible and effective.**

Keywords-component; AGC OFDM; power estimation; binary logarithms;

I. INTRODUCTION

Orthogonal Frequency Division Multiplexing (OFDM) is a special multicarrier modulation technique in which all the subcarriers are orthogonal, it can combat multipath fading and has high spectral utilization. In OFDM systems, the random addition of multiple subcarriers in time-domain makes a higher peak to average power ratio of transmitting signal. Therefore, a good Automatic Gain Control (AGC) algorithm is critical for OFDM system.

In the receiver, the level of incoming signal can vary over a wide dynamic range. AGC is employed to make sure the signal is not out of ADC dynamic range, we call it external AGC [1]. The internal AGC is also necessary .It can be used to hold the average power of baseband signal close to a desired level, to provide reasonable signal for the follow digital signal processing, and meet the processing precision. In this paper, we mainly introduce the digital internal AGC implement in FPGA.

The remainder of this paper is divided as follows. Section II has an overall view of the AGC architecture. Section III introduces the proposed design of AGC loop on FPGA. Section IV has the mathematical analysis of the AGC loop stability and the performance of the proposed design is shown. Section V gives a conclusion.

Agc model

In OFDM, it is generally believed that the channel is constant in a whole frame, so the whole frame has the same attention after channel transmission. In our OFDM system, each transmitted burst consists of a long training sequence (LTS) and eight data symbols as shown in Fig.1 [2]. The LTS consists of two short training sequences. The LTS is intended for AGC, equalization, and synchronization for purpose. In every frame, after the AGC has been locked, it can get an appropriate and stable control gain value to adjust the power of data symbols after the long training sequence. So the data symbols has a linear gain, it will not affect the follow digital signal processing.

The implement of digital AGC loop is shown as the follow Fig.2 [3]. The output signal passes the matched filter, which is power estimate, then compares to the reference power, bringing on the error signal. The error signal passes loop filter generates the conditioning signal, and then the conditioning signal is sent to adjust the input signal. There are two kinds of AGC loop [3]. One is based on a linear relation between the received signal and the reference and the other is based on the logarithmic of the received signal against the reference. In the linear relation, the steady state response of AGC loop is controlled by input signal, rather than loop gain constant α . It will generate slow system response or overshoot response. In view of this fact, we adopt the other one.

A simplified schematic diagram [3] of an digital AGC loop is shown in Fig. 2, where $x(n)$ represents input signal, $\log_2(.)$ is the logarithmic transformation and $2^{(.)}$ is the anti-log transformation, P_{ref} is the reference power, α is the constant loop gain, $A(n)$ the control gain, z^{-1} is the delay unit.

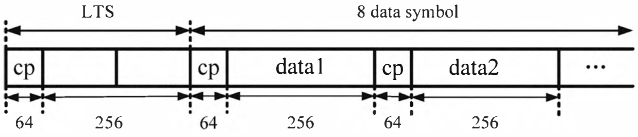


Figure 1. Baseband Framing

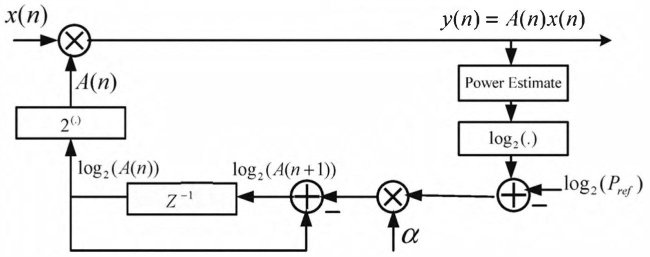


Figure 2. AGC Block Diagram

978-1-4244-7164-5/$26.00 © 2010 IEEE

II. Proposed Design

A. Implement of Power Estimate

For the signal $x(n)$ [4], there is a traditional method to calculate the average power, the formula is given by:

$$p(n) = \sum_{n=0}^{M-1} \{|\mathrm{Re}[x(n)]|^2 + |\mathrm{Im}[x(n]|^2\} / M \ . \qquad (1)$$

We can consider it as FIR format. M is usually set to 2^n, so the division operation can be changed into shift operation. Anther IIR format is expressed by:

$$p(n) = (1-\rho)p(n-1) + \rho\{|\mathrm{Re}[x(n)]|^2 + |\mathrm{Im}[x(n)]|^2\} \ . \qquad (2)$$

ρ is the weighting factor, it is also usually set to 2^{-n}. In performance, (2) has a longer response as IIR filter, it will be better than (1)[4]. In calculated amount, every M signals, (1) costs M adders and one shifter, (2) costs $2M$ adders and M shifters.

There also have some common formulas such as [6], I^2+Q^2 , $|I|+|Q|$ and $\max(|I|,|Q|)+\frac{1}{2}\min(|I|,|Q|)$ to calculate the instantaneous power, in which, $I = \mathrm{Re}[x(n)]$, $Q = \mathrm{Im}[x(n)]$. They have difference influence on AGC loop. $|I|+|Q|$ will bring on deviation, but it only needs two absolute operations. In conclusion, we propose an improved method expressed by:

$$p(n) = (1-\rho)p(n-1) + \rho\{|\mathrm{Re}[x(n)]| + |\mathrm{Im}[x(n)]|\} . \qquad (3)$$

B. Implement of Logarithmic Transformation in FPGA

Logarithmic transformation is usually implemented by look-up-table in hardware. The storage of a table of logarithms in FPGA would be inconvenient considering the memory capacity that would be required. Considering of FPGA area and power consumption, we adopt binary logarithms as logarithmic transformation on implementation of AGC model. In binary logarithms, it can conveniently use shift operation instead of involution and evolution operation on FPGA. Hence, to calculate the binary logarithm of a binary number, we need detecting the bit position of the most significant "one", then ignoring the most significant "one" and considering the bits to the right of most significant "one" as mantissa [5]. For example: $X = x_{n-1}x_{n-2}\cdots x_t \cdots x_0$, X is a n bits of binary number, x_t is the most significant "one", then we can get the $Y = \log_2 X = t + \log_2(1+m)$, in which, t is the integer part of logarithm of X , $m = 0.x_{t-1}x_{t-2}\cdots x_1x_0$ can be approximately consider as the fraction part.

The circuit of finding the position of the most significant "one" is the most important part in the binary logarithms, it can be implemented by combinational logic in FPGA, and the area would be saved. The specific technique is referred by [6].

The proposed algorithm can only assure the precision of integer part. It is usually used in the application which has rigorous demand on real-timing and hardware resource without calculation accuracy.

C. Calculation of Reference Power

The selection of the reference power makes sure the signal would not be overflow and meet the processing precision in the follow digital processing. For calculating the reference power, assuming the input signal achieves full-scale level, which is the maximum power Fs=0dB and the power of signal would be adjusted to -15dBFS (full-scale) by AGC. It is depended on the follow digital processing demand in the system. It can be expressed by:

$$10\log_{10}\frac{A^2}{\sigma^2} = 15dB \cdot \qquad (4)$$

σ^2 is considered as the power of AGC output signal, A is the full-scale level of input signal, which is normalized to 1. $A=1$ is substituted into (4) , we can obtain the $\sigma = 10^{-\frac{3}{4}}$.

The OFDM signal approximately obeys complex Gaussian distribution, that is $N(0,\sigma^2)$. In this paper, estimating power uses the (3) referred by Part A, and the estimated power is biased. The reference power is equivalent to reference level. The calculation of reference level as follow:
The PDF of $I = \mathrm{Re}[x(n)]$ or $Q = \mathrm{Im}[x(n)]$ is:

$$p(x) = (\frac{1}{2\pi\delta^2})^{1/2}\exp(-\frac{x^2}{2\delta^2}), \ (\delta^2 = 0.5 * \sigma^2). \qquad (5)$$

The expectation of $I = \mathrm{Re}[x(n)]$ is given by:

$$\begin{aligned} E(|I|) &= \frac{1}{\sqrt{2\pi}\delta}\int_{-\infty}^{+\infty} x\exp(-\frac{x^2}{2\delta^2})dx \\ &= \frac{2}{\sqrt{2\pi}\delta}\int_{0}^{+\infty} x\exp(-\frac{x^2}{2\delta^2})dx = \sqrt{\frac{2}{\pi}}\delta \end{aligned} , \qquad (6)$$

Because the PDF of I and Q is identical, we can obtain:

$$E(|I|+|Q|) = 2*\sqrt{\frac{2}{\pi}}\delta = \frac{2}{\sqrt{\pi}}\sigma \ \cdot \qquad (7)$$

From (4), we know $\sigma = 10^{-\frac{3}{4}}$, then the reference level is considered as $E(|I|+|Q|) = \frac{2}{\sqrt{\pi}}\sigma = 0.2 \cdot$

III. Aynalisis and Result

The performances of AGC loop can be analyzed from two aspects, system tracking time and gain jitter.

A. System Tracking Performance

For analyzing tracking time, The AGC model [4] is equivalent to the model as shown in Fig. *3* in Log-domain. The system is linear.

Assuming the power of signal can be estimated exactly, that is, the input P_{in} can be equivalent to step signal, the system step response can be derived from the model as follow:

$$G(n) = (1-\alpha)G(n-1) - \alpha(P_{in} - P_{ref}), \tag{8}$$

So the recursion can be expressed by:

$$G(n) = (P_{ref} - P_{in})[1-(1-\alpha)^n]u(n), \tag{9}$$

Then the signal P_{est} is obtained as:

$$\begin{aligned} P_{est}(n) &= P_{in} + G(n) \\ &= P_{in}(1-\alpha)^n u(n) + P_{ref}[1-(1-\alpha)^n]u(n) \end{aligned} . \tag{10}$$

From (10), if $0 < \alpha < 2$, the system is stable. Do a Z transformation on (10), we obtain:

$$\begin{aligned} P_{est}(z) &= \frac{P_{in}}{1-(1-\alpha)z^{-1}} + \frac{P_{ref}}{1-z^{-1}} - \frac{P_{ref}}{1-(1-\alpha)z^{-1}} \\ &= \frac{zP_{in}}{z-1+\alpha} + \frac{zP_{ref}}{(z-1)(z-1+\alpha)} \end{aligned} \tag{11}$$

According the final value theorem, the stable value of P_{est} is:

$$P_o = \lim_{Z\to 1}(z-1)P_{est}(z) = P_{ref}. \tag{12}$$

So we can get that the steady-state error satisfy the system demand.

Assuming $|P_{est}(n) - P_{ref}| < 1dB$, the AGC loop is locked, and then the tracking time is derived as follow:

$$|(P_{in} - P_{ref})(1-\alpha)^n| < 1 \tag{13}$$

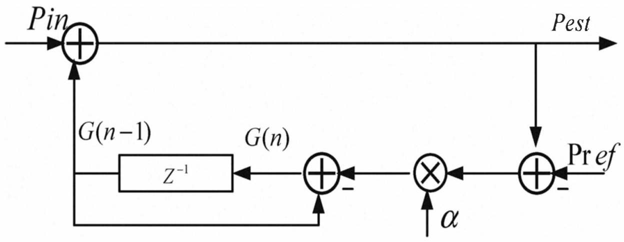


Figure3. AGC Model in Log-domain

Then we can get the setting time given by:

$$n < \frac{-\ln|(P_{in} - P_{ref})|}{\ln|1-\alpha|} \tag{14}$$

From (14), the AGC locked time relates to the loop gain and the input signal power. In our system, the length of the LTS is 256, so the tracking time n must be smaller than 256 sample points. Fig. 4 shows the relationships of loop gain and lock time.

B. Gain Jitter

For analyzing tracking time, we assume the power can be estimated exactly, but in fact, there is an estimation error which the jitter of gain generates from. The estimated power is expressed by $P_{est} = m_y + n_y$, m_y is the average power, and n_y is the estimating error. We believe that n_y is much smaller than m_y after logarithmic transformation,

$$\log_2(m_y + n_y) \approx \log_2 m_y + \frac{n_y}{m_y \ln 2}. \tag{15}$$

Assuming that m_y is a constant, n_y is considered just for the gain jitter. The analysis model [4] of gain jitter is shown in Fig. 5. From the model, we can get the transfer function of the gain jitter is:

$$\frac{\Delta G(z)}{N(z)} = \frac{\alpha}{z-1+\alpha} \tag{15}$$

[4] points out n_y is inappropriate to model as Gaussian white noise sequence.

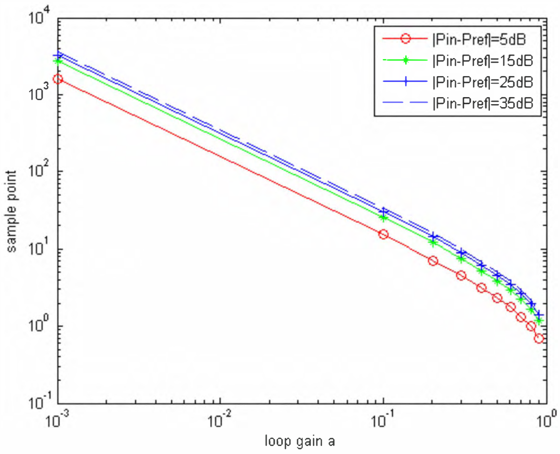


Figure4. Loop Gain & Lock Time (sample point)

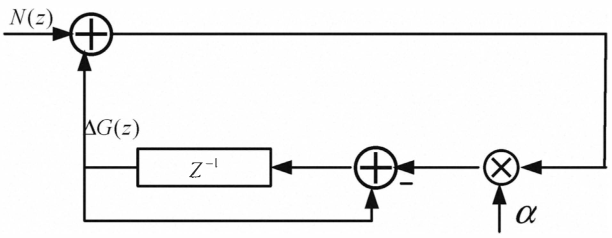


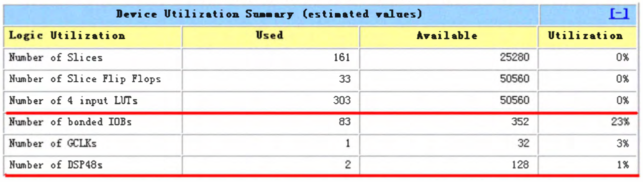
Figure5. Analysis Model of Gain Jitter

C. Achieving Result on FPGA

The AGC design was implemented on Xilinx Virte4 FPGA Device XC4VFX60, in which the frequency of the system clock was $f_{clk} = 120M$ and the sample rate was $f_s = 6M$. The TABLE I is cut from ISE Design Summary. As we can see from TABLE I, only less than 1% of LUTs and two DSP48s are used by one AGC.

The Fig. 6 and Fig. 7 show the result of AGC implementation on FPGA. All of the input signals and output signals were sampled from FPGA by chipscope. As we can see from the Fig. 6 and Fig. *7*, the input signal waves on the top of figures can change in a wide dynamic range. After AGC, they can hold in a fixed level which is shown in the two waves on the bottom of figures.

TABLE I IMPLEMENTATION RESULT



Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	161	25280	0%
Number of Slice Flip Flops	33	50560	0%
Number of 4 input LUTs	303	50560	0%
Number of bonded IOBs	83	352	23%
Number of GCLKs	1	32	3%
Number of DSP48s	2	128	1%

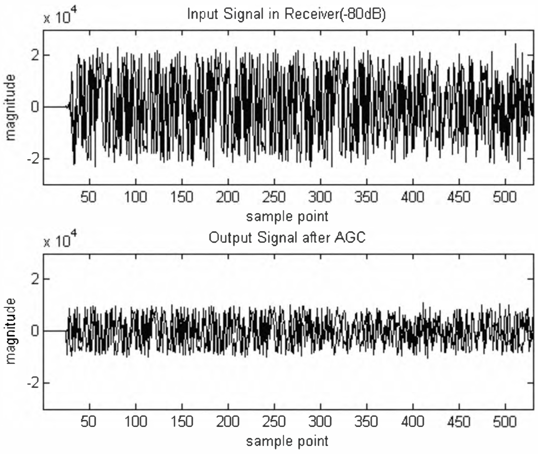


Figure 6. Testing Result 1

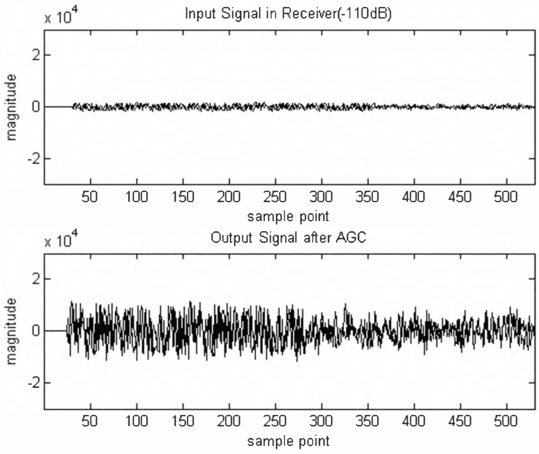


Figure 7. Testing Result 2

IV. CONCLUSION

In this paper, we study the principle of the AGC loop, and propose an AGC loop model. On this basis, we give a detail on how to design parts of AGC loop on FPGA, including implementation process and setting parameters. The performances of locking time and gain jitter are given in theory. The implementation on FPGA has been tested in OFDM system.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments and suggestions. This work is supported by a grant Natural Science Foundation of Fujian Province, P.R.China (No.A0710021) and the Major Science and Technology special project of Fujian Province, P.R.China (No.2009HZ0003-1).

REFERENCES

[1] Lei Li, Changgen Zhu, "On AGC Design and Digital Realization in All-digital-receiver", Journal of Chongqing University of Science and Technology, Vol.10, pp.97-99, Jul.2008

[2] Meifen Chen, "Design and Implementation of Baseband on Broadband Wireless Ad Hoc Network", unpublished.

[3] Chao Wang, Yu Zongying, "Analysis and design a structure of fast digital AGC", Computer & Information Technology, Vol.3, pp.41-43, 2008.

[4] Jian Sun, "Study on Synchronization for OFDM System in Burst Mode", unpublished.

[5] John N.Mitchell, "Computer multiplication and division using binary logarithms", IRE Trans on electronic computers, Vol.8, pp.512-517, Aug.1962.

[6] Shengmei Mou , Xiaodong Yang, "Design and Implementation of an Improved FPGA-Based 32-Bit Logarithmic Converter", Journal of Computer Research and Development, Vol.7, pp.1252-1257, 2007.