Research Based on IEEE1588 Frequency Compensation Algorithm

Xiao Liu¹ and Changjun Li²

¹Guilin University of Electronic Technology School of Electronic Engineering and Automation, Guangxi Guilin, 541000
²Guilin University of Electronic Science and Technology, Guangxi Guilin, 541000

Abstract. The paper first studies the IEEE1588 network clock synchronization system. Aiming at the instability of the system clock crystal oscillator, a frequency compensation algorithm is proposed, which can adjust the master-slave clock frequency and ensure the master-slave clock synchronization. The simulation analysis shows that when the master-slave clock generates the deviation value, the master-slave clock is consistent with the execution of the frequency compensation algorithm, which verifies the feasibility and effectiveness of the algorithm.

1. Introduction

The full name of IEEE1588 is "IEEE1588 Precision Clock Synchronization Protocol [1]". The IEEE 1588 standard is a general specification for improving the timing synchronization capability of network systems, enabling distributed communication networks to have strict timing synchronization and being applied in various systems. At present, the clock precision achieved by traditional clock synchronization protocols such as NTP and simple network time protocol can only reach millisecond level [2], and the IEEE1588 protocol [3-5] improves the clock synchronization accuracy. Make clock accuracy to the microsecond level.

The article specifically introduces the basic principle of the IEEE1588 clock synchronization system [6], for the effect of crystal stability on clock accuracy, on this basis, the frequency compensation algorithm is proposed to achieve the stability of the master-slave clock crystal frequency, and finally, the feasibility of the algorithm is verified by simulation.

2. IEEE1588 protocol clock synchronization principle

In the IEEE1588 clock synchronization process, the master clock and the slave clock are first determined by the best master clock algorithm [7], and then the time offset of the two clocks and the transmission delay of the message are determined by exchanging messages. The protocol mainly defines four types of clock messages: synchronous message Sync, follow message Follow_Up, delayed request message Delay_Req, and delayed request response message Delay_Resp. Figure 1 shows the process of IEEE1588 protocol implementation. The process of clock synchronization is divided into two phases [8]:

Clock offset measurement: The master clock node sends Sync to the slave clock every 2s. This information includes the estimated value of the master clock transmission time. After the transmission, it then sends Follow_Up, which contains the Sync accurate transmission time t1. The slave clock node
records the reception time $t_2$ after receiving the Sync message. At this time, the time offset Offset between the master and slave clock nodes can be calculated by $t_1$ and $t_2$. 

Clock delay measurement: Send a third message Delay_Req from the clock node to the master clock node, and record the message transmission time $t_3$. After receiving the Delay_Req, the master clock node records the reception time $t_4$, writes $t_4$ to Delay_Resp, and finally sends the message to the slave clock. Calculate clock delay based on $t_3$ and $t_4$. 

![IEEE1588 protocol synchronization principle.](image)

Ideally, the message information sent by the master clock node to the slave clock node and the message information sent from the clock node to the master clock node is the same, that is, the two Delays in the figure are the same, then:

\[ t_2 = t_1 + O + D, t_4 = t_3 - O + D \]  

(1)

The Delay and Offset can be calculated by the formulas (1), as follows:

\[ D = \frac{(t_2 - t_1) + (t_4 - t_3)}{2}, O = \frac{(t_2 - t_1) - (t_4 - t_3)}{2} \]  

(2)

After calculating $D$ and $O$, adjust the slave clock node according to these two values to complete a clock synchronization process. When the deviation occurs again, repeat the steps.

3. **IEEE1588 protocol frequency compensation algorithm**

For the problem of crystal stability, the proposed frequency compensation algorithm periodically detects the time error. When the master-slave clock generates a deviation value due to the different crystal oscillator frequency, the signal sampling time is adjusted, and the clock frequency is adjusted by adjusting the delay of the sampling point or sampling in advance. The purpose of compensation. This algorithm takes about three steps:

Step 1: Assuming that the sending time is $T_{ms}$, the receiving time is $T_{si}$, and the sending and receiving delay is Delay, then the corresponding time deviation is:
\[
\text{Offset}_i = T_{si} - T_{mi} - \text{Delay}
\]

Then, after $K$ times, the cumulative amount of deviation can be reached.

\[
t = \sum_{i=1}^{k} \text{Offset}_i = \sum_{i=1}^{k} (T_{si} - T_{mi} - \text{Delay})
\]

In Equation 4, the sending time is $T_{mi}$, and the receiving time is $T_{si}$, all available through the system. Then the calculation formula 4 only needs to determine that the system sending and receiving delay is $\text{Delay}$.

Suppose that in the IEEE1588 synchronization system, the transmission period of the Sync message is set to $T_1$, in addition, the transmission delay defaults to 2 seconds; then the transmission period of the Delay_Req message is set to $T_2$, and the transmission delay is defaulted to $2R+Q/2$ second, set $T_2$ between 4 seconds and 60 seconds; the delay from master clock to slave clock and slave clock to master clock are set to $\text{MS\_delay} = T_{s\_m} - T_{m\_s} - \text{offset}$, $\text{SM\_delay} = T_{m\_s} - T_{s\_m} + \text{offset}$ respectively; then the actual network delay can be expressed as:

\[
\text{Delay} = \frac{2}{2+2R+Q/2}(T_{s\_m} - T_{m\_s}) + \frac{2R+Q/2}{2+2R+Q/2}(T_{m\_s} - T_{s\_m})
\]

In Equation 5, the parameters $Q$ and $R$ can select any value, but considering the accuracy of the FPGA implementation and the limitation of the bit width, the $R$ value can be set to 2, and the $Q$ value is 4, then the formula 5 can be expressed as:

\[
\text{Delay} = \frac{1}{4}(T_{s\_m} - T_{m\_s}) + \frac{3}{4}(T_{m\_s} - T_{s\_m})
\]

After $K$ times, the cumulative amount of deviation can be expressed as:

\[
t = \sum_{i=1}^{k} \left[ T_{si} - T_{mi} - \left( \frac{1}{4}(T_{s\_m} - T_{m\_s}) + \frac{3}{4}(T_{m\_s} - T_{s\_m}) \right) \right]
\]

Step 2: The decision is made according to the value obtained by 7 and the preset threshold. If it is greater than the threshold, then the process proceeds to step 3, otherwise step 1 is repeated.

Step 3: When the accumulated amount of deviation reaches the threshold, the system generates a clock reset signal for correcting the sampling time of the clock, clearing the accumulated amount of deviation.

4. Simulation Analysis
According to the frequency compensation algorithm, it is simulated and analyzed by ModelSim software. The standard frequency is 50MHz, and two different clocks are set, one of which has a clock frequency of 50KHz and the other clock frequency of 49.95KHz. After the frequency compensation algorithm is executed, the simulation results are shown in Figure2.
The primary clock is verified. As can be seen from Figure 3, o_clk1 and o_clk2 are the master-slave clocks generated by the configuration. Because the frequencies of o_clk1 and o_clk2 set above are different, there will definitely be some deviation between the two clocks. O_clk2n represents the clock modified by the frequency compensation algorithm. From the above simulation results, it can be seen that the corrected o_clk2n and o_clk1 are basically synchronized, and the o_clk2 clock and o_clk1 have large deviations. In order to better see the effect of clock frequency compensation, the data in the red box above is enlarged, so that the exact time value before and after clock compensation can be seen more clearly, as shown in Figure 3.

As can be seen from Figure 3, o_clk1 and o_clk2 have a lot of difference at this time. After measuring with the cursor, it can be concluded that the two clocks have a difference of 4120 ns, the deviation is relatively large, and the time difference between the measurement of o_clk1 and o_clk2n is measured. After that, it was found that the deviation of the measurement at this time was only 100 ns, and the accuracy was relatively high.

5. Summary
As can be seen from the above. The stability of the crystal oscillator is an important factor to improve the synchronization accuracy. For this problem, the frequency compensation algorithm is proposed in this paper. The simulation is implemented according to this algorithm, and the feasibility of the algorithm is verified.

References
[1] Sven Meier, Hans Weibel, Karl Weber. IEEE 1588 Syntonization and Synchronization Functions Completely Realized in Hardware [J]. ISPCS, 2008.
[2] Du Huiqi, Liu Jue, Liu Bo, etc. Application of IEEE1588v2 clock synchronization technology in tactical information system [J]. Computer application and software, 2017, 11:125-129.
[3] CROSSLEY P A, GUO H, MA Z. Time synchronization for transmission substations using GPS and IEEE1588 [J]. CSEE Journal of Power and Energy Systems, 2016, 2(3):91-99.
[4] LEE S. An Enhanced IEEE1588 Time Synchronization Algorithm for Asymmetric Communication Link Using Block Burst Transmission [J]. IEEE Communication Latters. 2008, 12(9):687-689.
[5] Research and Implementation of Xing Yongping. IEEE1588 Precision Clock Synchronization Protocol [D]. Beijing Jiaotong University, 2013.
[6] Zdenek Chaloupka, Nayef Alsindi, James Aweya. Efficient and Precise Simulation Model of Synchronization Clocks in Packet Networks [J]. CAMAD, 2013:79-83.
[7] SCAGLIA S. The Embedded Internet TCP/IP Basics, Implementation and Applications [M]. New Jersey: Addson Wesley. 2008:15-37.
[8] GURUSWAMY A, BLUM R S, KISHORES, et al. Performance lower bounds for phase offset
estimation in IEEE1588 synchronization [J]. IEEE Transactions on Communications, 2015, 63(1):243-253.

[9] Zdenek Chaloupka, Nayef Alsindi, James Aweya. Efficient and Precise Simulation Model of Synchronization Clocks in Packet Networks [J]. CAMAD, 2013:79-83.