ISSN 2959-6157

Design and Implementation of Digital Integrated Circuit Based Frequency Dividers

Zirui Zhang

Southampton University, Southampton, USA
*Corresponding author: zzr20050722@gmail.com

Abstract:

This research aims to research the basic fractional frequency division. This article first introduces the theory of the dual-mode prescaler method, and then use this method to calculate the number of integer frequency divisions and show that how to achieve the target fractional division ratio. after that, Verilog code is used to implement fractional frequency division, and using simulation to show the operation of the divider. The simulation shows that the designed fractional frequency divider can be implemented by evenly inserting different values of integer frequency dividers. However, the experiment in this article may has time error in the output signal, because the accurate 50% duty cycle cannot be achieved. In addition, the frequency division period is not completely uniform. Finally, the challenges and limitations of fractional frequency division technology in theory and practice will be summarized.

Keywords: fractional frequency division; digital integrated circuit; simulation experiment.

1. Introduction

The integrated circuit industry was keep evolving over the past few decades, particularly the advances in transistor miniaturization and the integration of multiple functions onto a single chip. In the post-Moore era, the focus of integrated circuit technology has shifted from simple size reduction to improving power efficiency, performance, and overall system integration. One of the important technological breakthroughs was the development of multi-gate transistors which provided better performance and lower power consumption. In addition, the emergence of new materials and packaging technologies has also enabled integrated circuits to have better capabilities compared with traditional silicon-based technologies [1]. Moreover, with Moore's Law reaching the physical limit, the traditional planar transistor structure has been unable to satisfy the requirements of further shrinking size and lower power consumption. Therefore, there are increasing number of new technologies, such as FinFET (fin-type field effect transistor) technology, which will become the mainstream of current CMOS technology [2].

PLL is a circuit widely used in modern power electronic equipment. The primary function of PLL is to detect the phase angle of an input signal and use closed-loop feedback to lock onto that phase angle, ensuring that the output signal remains synchronized with the input signal. In other words, PLL includes phase detectors, controllers, and a feedback loop to provide fast and stable phase tracking capabilities in a variety of applications. For example, grid synchronization: In grid-connected inverters, PLL is used to synchronize with grid voltage for correct power and reactive power injection [3].

The frequency divider is a classic digital integrated circuit, and it is a very importamt part in the phase-locked loop (PLL), especially it can generate high-precision signals in cases of high frequency and low power consumption conditions [4]. In addition, with the advancement of 5G technology, the application of frequency dividers in the millimeter wave band has become the key point. In the near future, the design of frequency dividers will tend to be more integrated and smaller chip sizes to adapt to the design needs of modern electronic products [5].

There are many kinds of dividers, and the decimal divider is one of them. Precise frequency control by adding a variable frequency divider to the integer frequency divider to achieve higher frequency resolution [6]. This article will explain the logic and implementation of fractional frequency division in detail. In addition, the detailed process of fractional frequency division is shown through Verilog code and finally the simulation results are used to explain the fractional frequency division.

The first part of the thesis is the introduction which introduces the research background and significance of the thesis. Chapter 2, method will give details of working logic and implementation of fractional frequency division. The

third chapter is results, which introduces how to know the accuracy of experimental results by testbench and simulation. The defects of the experiment are given in Chapter 4. The last part is the conclusion, which summarizes the whole paper.

2. Method

2.1 Basic formula

Before the code operation, it is necessary that understanding how to complete the fractional frequency division by using the formula of dual-modulus prescaler method [7].

$$N_{frac} = \frac{N * X + (N + S) * Y}{X + Y} \quad (1)$$

Irregular fractional division cannot achieve fractional division multiple of the source clock cycle after each clock cycle, such as 5.4 fractional division. For this kind of non-integer frequency division, it cannot be obtained directly through the integer frequency division circuit, because the integer frequency division can only produce integer multiple frequency division results. However, this problem can be solved by dual-modulus prescaler method. The dual-modulus prescaler method approximates the target fractional division ratio by alternating between two different integer division factors. For example, to achieve a division ratio of 5.4 can alternate between integer frequency divisions of 5 and 6.

N_{frac} is the fractional division ratio in the formula. In the above example, it is equal to 5.4. X is the number of times of frequency division of N is used, and Y is the number of times of frequency division of N+S is used. N is the value of lower integer frequency divisions, and (N+1) is the value of higher integer frequency divisions. According to the reference, S is the correction term, and S is often equal to 1.

The first step for using the formula is to determine the value of N. In this case, let N = 5. When a fractional frequency divider is desired (e.g., 5.4 times), two integer frequency division values, M and M+1, need to be found such that the combination of these two values approximates the target value. The closest integers to the target value are 5 and 6, so the choice of N=5 is based on its proximity to the target of 5.4.

Bringing N=5 and N_{frac}=5.4 into Equation:.

$$5.4 = \frac{5 * X + 6 * Y}{X + Y} \quad (2)$$

$$X + Y = 10 \quad (3)$$

Finally, the results, X=6 and Y=4, satisfied the target division ratio requirement.

2.2 Distribution of frequency divisions

After using those equations, consider the order of different integer frequency divisions to Implement fractional frequency division signals. There are two methods. First method, 6 division 4 times followed by 5 division 6 times, or performing 5 division 6 times followed by 6 division 4 times. However, the sudden change in the clock period will lead to large phase jitters which has negative impact on the stability of signal processing of subsequent circuits. The second method, evenly inserting 6 divisions of 5 between 4 divisions of 6, or evenly inserting 4 divisions of 6 between 6 divisions of 5. This alternating approach between different multiplication factors can smooth out the variations in the clock cycle.

In fractional frequency division operations, the magnitude of the difference determines that which frequency division (shorter or longer) should be selected. There is important mathematical and temporal logic behind the choice.

After using those equations, consider the order of different integer frequency divisions to Implement fractional frequency division signals. There are two methods. First method, 6 division 4 times followed by 5 division 6 times, or performing 5 division 6 times followed by 6 division 4 times. However, the sudden change in the clock period will lead to large phase jitters which has negative impact on the stability of signal processing of subsequent circuits

When the difference is large (greater than 10), it means that there is large difference between the current cumulative number of time cycles and the target value. If continue to use the 5-division frequency, it may lead to the cumulative number of periods beyond the target. At this time, choosing 6 divisions can decrease the gap quickly, and make the cumulative period count closer to the target.

In addition, the 6-frequency division has an extra period compared to the 5-frequency division. 6-frequency division can adjust the value of difference in a large extent, so that the accumulated number of periods quickly approaches the final target. Therefore, it can avoid too large or too small differences.

When using 5-division frequency, assuming the current cumulative difference is d. The next cumulative difference will be increased by 4 (54 - 10 x 5), because the 5-division frequency increases the number of cycles by 5 at a time, whereas the target is 10 cycles at a time.

$$d_{new} = d + (54 - 10 \times 5),\ d_{new} = \mathrm{d} + 4 \quad (4)$$

When using 6-division frequency, the next cumulative difference will be reduced by 6 (54 - 10 x 6) because 6-division frequency increases the number of periods by 6 each time.

$$d_{new} = d + (54 - 10 \times 6),\ d_{new} = \mathrm{d} - 6 \quad (5)$$

For some examples, first division: Initial difference: 54 - 10×5 = 4. Since the difference is less than 10, choose 5 divisions. Second division: Accumulate the difference: 4 + 4 = 8. The difference is still less than 10, so continue to choose 5 divisions. Third division: Accumulate the difference: 8 + 4 = 12. The difference is greater than 10, so choose 6 frequency division. Fourth division: Update the difference: 12 - 6 = 6. The difference is less than 10, so choose 5 division. And so on.

2.3 Divisions code implementation

```
//Fractional frequency divider circuit design
//dual-modulus prescaler method to achieve 5.4 frequency division
module clk_div_fraction
    (
    input               rst_n,      //Reset Signal
    input               clk,        // Clock signals
    output              clk_frac    // Fractional frequency division output sign
    );

 //Define the 5.4 frequency division for 5 and 6 frequency division
parameter       CLK_DIV_1  =  5;
parameter       CLK_DIV_2  =  6;
parameter       DIFF       =  4;    //Difference between 5 crossover frequency and 5.4 crossover frequency in 10 cycles


reg [3:0]           cnt_end;          // frequency division insertion counter (used to determine what frequency division to insert)
reg [3:0]           cnt;              // Total counter
reg                 clk_frac_r;       //Fractional division intermediate register signal
reg [4:0]           diff_cnt_r;       //Difference signal
reg [4:0]           diff_cnt;         //Difference signal
wire                diff_cnt_en= cnt == cnt_end;       //enable signal


//Difference accumulation logic module
always @(*) begin
    if(diff_cnt_r >= 10) begin
        diff_cnt = diff_cnt_r -10 + DIFF;   //If the difference is greater than 10, insert 6 frequency division and subtract 6 from the difference


    end
    else begin
        diff_cnt = diff_cnt_r + DIFF;       //If the difference is less than 10, insert 5 frequency division and add 4 to the difference
    end
end

// Borrowed register delayed output diff_cnt_r
always @(posedge clk or negedge rst_n) begin
    if(!rst_n) begin                //Reset the difference to zero
        diff_cnt_r <= 0;
    end
    else if(diff_cnt_en) begin      //Enable the difference signal to delay output at high voltage
        diff_cnt_r <= diff_cnt;
    end
end
```

Fig. 1 Verilog code implement (the top half part)

```

// modules: The 5 and 6-division logic inserted
always @(posedge clk or negedge rst_n) begin
    if(!rst_n) begin
        cnt_end <= CLK_DIV_1-1 ;        // Insert 5 division first if Reset
    end
    else if(diff_cnt >= 10) begin
        cnt_end <= CLK_DIV_2-1 ;        //If the difference is greater than 10, insert 6
    end
    else begin
        cnt_end <= CLK_DIV_1-1 ;        //If the difference is less than 10, insert 5
    end
end

//
always @(posedge clk or negedge rst_n) begin
    if(!rst_n) begin
        cnt <= 1'b0;
        clk_frac_r <= 1'b0;
    end
    else if(cnt == cnt_end) begin       //Counter to  Boundary Point of frequency division Insertion
        cnt <= 1'b0;                    //Total counter cleared to zero
        clk_frac_r <= 1'b1;             //Clock division level set to "1"
    end
    else begin          //In other cases, the counter accumulates and the clock division signal level remains "0".
        cnt <= cnt + 1'b1;
        clk_frac_r <= 1'b0;
    end
end

//延时输出，消除亚稳态
assign clk_frac = clk_frac_r;

endmodule
```

Fig. 2 Verilog code implement (the bottom half part)

Fig. 1 and Fig. 2 show the Verilog code.

From the Fig.2, In 5 and 6 frequency insertion logic modules, cnt_end represents a counter maximum of cnt. Counter counting period from 0 to cnt_end. When cnt_end is 4, the counter will count from 0 to 4 for a total of 5 clock cycles. Therefore, 4'h4 corresponds to the frequency division operation of 5 clock cycles, that is 5 frequency division

3. Result

3.1 Testbench

```
`timescale 1ns/1ps
module clk_div_fraction_tb;


reg         clk;
reg         rst_n;
wire        clk_frac;

parameter DIV_CLK = 5;


//the reset signal is produced
initial begin
    clk = 0;
    rst_n = 1;
    #(3*DIV_CLK)
    rst_n = 0;
    #(6*DIV_CLK)
    rst_n = 1;
    #(20*DIV_CLK);
end


always #DIV_CLK clk = ~clk;



initial begin

    $display("Simulation started");

    $monitor("Time: %0t | clk = %b | rst_n = %b | clk_frac = %b", $time, clk, rst_n, clk_frac);
end


clk_div_fraction u_clk_div_fraction
    (.clk             (clk),
     .rst_n           (rst_n),
     .clk_frac        (clk_frac)
    );

endmodule

```

Fig. 3 Testbench code

From Fig. 3, parameter DIV_CLK = 5; means that set the period of a source clock signal to five-time units.

always #DIV_CLK clk = ~clk; means that after every 5-time unit, the clock signal is flipped once, which can generate a clock signal with a period of the 10-time unit.

In the part of 'the reset signal is produced', the clock signal is initialized by low, and the reset signal is initialized by high (non-reset state). After 3 clock cycles, the reset signal is pulled low to trigger a reset; after 6 clock cycles, the reset signal is high to release the reset and then waits for 20 clock cycles.

3.2 simulation results

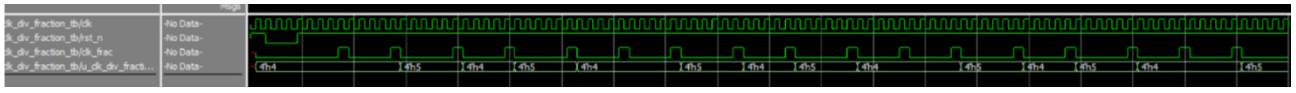

Fig. 4 simulation results

The Fig. 4 is simulation results which shows the four most important signals.

The first one is clk (input clock), it is the input source clock signal, and it is the reference clock of the system. According to the Verilog code, each rising edge of the clock signal will trigger the frequency divider logic, and the output of clk_frac is based on the counting and dividing result of this signal.

The second is rst_n (reset signal), the low level means reset. In the waveform, the signal starts low and then turns to high (normal operating state). According to the Verilog code, during a reset, the divider is reset, and all counters and registers are cleared to zero. When the rst_n signal goes high, starts operating.

The third signal is clk_frac (the fractional division output signal). This is the output clock signal after 5.4 frequency division. The period keeps changing, each period corresponding to a certain number of clk signal cycles. In the waveform diagram, the value of the cnt_end signal changes from 4'h4 (indicating 5 cycles) to 4'h5 (indicating 6

cycles), which indicates that the circuit is alternately performing 5 and 6-division frequency. With this alternative waveform, the circuit achieves 5.4 frequency division, i.e., for every 10 output cycle clocks completed, 54 clock cycles of the input have been exhausted.

The last one is cnt_end. The cnt_end determines the termination value of the current frequency division cycle, and it alternates between 5 and. In the waveform, cnt_end signal alternates between 4‘h4 and 4’h5, which means that the counter performs 5 or 6 counts in different cycles. That is exactly the logic needed to implement fractional frequency division.

This simulation proves the successful implementation of 5.4 fractional frequency. The waveform graph clearly shows the switching of the frequency division between 5- and 6-frequency division, which satisfies expected behaviours.

4. Discussion

When arranging integer frequency division, there are some problems if used the first way. Taking this experiment as an example, it will lead to the uneven clock signal and the phase jitter. 5.4 frequency division is combined by alternating 5 frequency division and 6 frequency division. If the clock signal is not uniform (the clock period is unstable), it may lead to the output clock signal period is unstable, resulting in greater timing errors [8].

Phase jitter can be reduced. The multimode integer frequency divider (MMD), ΣΔ modulator, and pipelined phase interpolator (PI) are combined to form an integrated system to solve the phase jitter problem [9].

In the simulation diagram, 5.4 fractional division does not have a uniform length for each segment (the local fraction does not meet the fractional fraction, and the overall fraction meets the fractional fraction). Because the period of the output clock will be alternating with the 5 and 6 divisions, rather than each cycle strictly following 5.4 times frequency division. Therefore, in a specific local period (such as 10 cycles), the signal cycle length will be different, and the uniform 5.4 times frequency division effect cannot be completely achieved.

In addition, in the experiment, there is the problem of duty cycle which directly affects the symmetry of the clock signal. If the period of high and low levels is uneven, it can lead to an advance or lag of the clock edge, which can cause timing errors. In this experimental example, each segment is not a strict 5.4 frequency division (because the signal flip-flop is only triggered at the edge), and the period is difficult to guarantee. More specifically, 5.4 times the frequency division is achieved by alternating 6 times the 5-frequency division and 4 times the 6-frequency division. The output period of 5 division frequency is shorter than that of 6 division frequency. When these unequal length cycles appear alternately, the high- and low-level time of the whole signal is difficult to reach the ideal 50%. Eventually, the duty cycle is not 50%, which may result in the clock signal cannot drive subsequent circuits evenly. And leads to data loss or transmission errors [10].

In the simulation diagram, 5.4 fractional division does not have a uniform length for each segment (the local fraction does not meet the fractional fraction, and the overall fraction meets the fractional fraction). Because the period of the output clock will be alternating with the 5 and 6 divisions, rather than each cycle strictly following 5.4 times frequency division. Therefore, in a specific local period (such as 10 cycles), the signal cycle length will be different, and the uniform 5.4 times frequency division effect cannot be completely achieved.

However, from view of longer period of time (e.g., 54 input clock cycles), the average crossover ratio of the output signal is close to 5.4 by judiciously arranging different intefer frequency division. This approach achieves an overall fractional frequency division by balancing the number of frequency divisions on the whole.

5. Conclusion

This paper takes 5.4 frequency division as an example to explain the working principle and implementation of fractional frequency division in detail. Firstly, through the formula dual-modulus prescaler method, to solve the problem that the irregular fractional frequency division cannot be achieved after the frequency division of each clock cycle is the fractional frequency division times of the source clock cycle. Using the formula to achieve a known frequency division target value.

After that, two methods of distributing frequency division are explained in detail. After comparing the effects of the two methods mentioned in this paper, the choice was made to alternate the averaging of the different crossovers, resulting in a smoother variation of the clock period. In addition, though the size of the difference to determine whether to use a shorter or longer frequency division. Based on the above formulae and mathematical logic, the fractional frequency division is achieved. Finally, the fractional frequency division is verified by Verilog code and simulation.

After the image of the simulation result is obtained, the analysis is carried out. The most important is the cnt_end, because cnt_end determines the end value of the current frequency dividing period. It can be seen from the waveform that the cnt_end signal alternates between different frequency divisions, which shows that the experimental

results are correct.

However, there are some shortages in this experiment. For example, it is impossible to reach the 50 percent duty cycle, which can lead to data loss or transmission errors in the real world.

In addition, there is the fact that in the simulated graphs, the 5.4 fractional frequency divisions are not of uniform length in every segment, so a uniform 5.4 fractional frequency division cannot be fully achieved in a particular localised period. The above points make this experiment only get the overall fractional frequency division, which is not entirely very accurate.

References

[1] Wang, Y., Chi, M.-H., Jesse Jen-Chung Lou, & Chen, C.-Z. (2023). Handbook of Integrated Circuit Industry (pp. 3–24). Springer Nature.

[2] Arai, K., Kapoor, S., & Bhatia, R. (2020). Proceedings of the Future Technologies Conference (FTC) 2020, Volume 2 (pp. 340–346). Springer Nature.

[3] Mohamed Samy El-Daleel. (2023). A Fully Linear PLL for Power Electronics Applications and PLL Tuning Guidelines Along with Comparative Study of Different Phase Detection Methods. 2023 IEEE Conference on Power Electronics and Renewable Energy (CPERE). IEEE Xplore. https://doi.org/10.1109/cpere56564.2023.10119579

[4] Zheng, Y., & Chen, X. (2023). A Low-Power RF programmable frequency divider. https://doi.org/10.1109/icccs57501.2023.10150629

[5] Dong, C., Dong, G., & Hu, S. (2020). A 5G Millimeter Low Power Phase-Locked Loop in Low-Cost 0.18-μm CMOS. 2020 IEEE MTT-S International Wireless Symposium (IWS). https://doi.org/10.1109/iws49314.2020.9360063

[6] Best, R. (2003). Phase-Locked Loops (pp. 103–107). McGraw Hill Professional.

[7] Cheung, T. H., Ryynanen, J., Parssinen, A., & Stadius, K. (2021). A 5.4-GHz 2/3/4-Modulus Fractional Frequency Divider Circuit in 28-nm CMOS. 2021 IEEE International Symposium on Circuits and Systems (ISCAS). https://doi.org/10.1109/iscas51556.2021.9401405

[8] Nicola Da Dalt, & Sheikholeslami, A. (2018). Understanding Jitter and Phase Noise (pp. 111–120). Cambridge University Press.

[9] Cheung, T., Martelius, M., Antonov, Y., Akbar, R., Ryynänen, J., Pärssinen, A., & Stadius, K. (n.d.). A 3.5-GHz Digitally-Controlled Open-Loop Fractional-N Frequency Divider in 28-nm CMOS.

[10] Zheng, Y., & Chen, X. (2023). A Low-Power RF programmable frequency divider. https://doi.org/10.1109/icccs57501.2023.10150629