Sensors and Materials, Vol. 35, No. 7 (2023) 2653–2668
MYU Tokyo

S & M 3346

An Auto Offset Calibration Method for High-resolution Continuous CMOS Comparators

Wen Guanguo,[1] Long Qiang,[2] Hu Huiyong,[1*] and Zhang Jincheng[1]

[1]School of Microelectronics, Xidian University,
No. 2 Taibai South Road, Xi'an 710071, China
[2]Suzhou Bosi Semiconductor Co., Ltd.,
Room D306, Building D, Block 6, No. 2888 Wuzhong Avenue, SuZhou 215000, China

(Received March 21, 2023; accepted July 18, 2023)

Keywords: comparator, auto offset calibration, high resolution

High-precision, continuous analog comparators are widely used in signal detection, alarm protection, and other fields. An auto offset calibration method for high-resolution continuous CMOS comparators (CMPs) was proposed. On the basis of the first output of the short-input format CMP, calibration logic will select the proper routine to calculate the best fix trim bit. Two calibration codes are added and averaged to obtain the actual code. This mainly takes into account the fact that there may be a certain delay in comparator flipping, which leads to deviations from the optimal calibration code. This part of the search error can be counteracted by averaging the results of searching from low to high and from high to low. According to different design needs, different trim step sizes can be obtained by adjusting the relative ratio of the smallest calibratio N-channel metal-oxide-semiconductor (NMOS) to the main input pair. Circuit implementation is based on the 110 nm flash process with a 5 V IO device. Analysis and simulation results show that a less than 1 mV offset can be easily achieved, which is suitable for commercial use. The proposed auto offset calibration method does not increase current consumption and can be easily shifted to other advanced technology processes, which make it promising for future use.

1. Introduction

With the evolution of modern lives, there is a high demand for portable, battery-operated, and low-power devices in the area of medical systems, wireless sensor networks, and customer electronics.[1–6] High-precision analog comparators are widely used in signal detection, alarm protection, and other fields. Commonly, the comparator is used as a critical block to trigger an event reporting to a top-level system.[7–12] It senses the voltage difference between two inputs and converts analog signals into digital outputs. Many factors will affect a comparator design, which decide the speed, power, signal-to-noise ratio, and bit error rate of the system. One of the most important specifications of the comparator is its offset voltage (V_{OS}). In many types of circuit, the comparator offset imposes a fundamental limit on the achievable performance.

*Corresponding author: e-mail: huhy@xidian.edu.cn
https://doi.org/10.18494/SAM4401

ISSN 0914-4935

Offset voltages often determine the level of signals that can be processed.[13–17] Offset voltages can be divided into system offset and random offset. System offset is generally related to architecture and design, whereas random offset is closely related to device area and layout.[18] Usually, the random offset is proportional to the reciprocal of the square root of the area. The larger the area, the smaller the offset. However, it is not good to increase the area indefinitely, so the accuracy of an ordinary commercial continuous analog comparator is usually 5 mV.[19] For applications with an accuracy of less than 1 mV for continuous comparison, it is difficult to achieve the design accuracy simply by increasing the area. In applications such as analog-to-digital converters (ADCs), where a clock is naturally used in signal sampling and control logic timing, one can use different clock phases to control the performance of a dynamic comparator to achieve both high speed and low offset. However, in certain applications such as utilizing a general-purpose comparator in MCUs, clocks may not be available when the CMOS comparator (CMP) is in operation. In these cases, the CMP may be used as an interrupt monitor to detect some internal or external continuous analog signals such as power supply or external-input voltage to trigger an interrupt event to the CPU.[19] Thus, reducing the offset voltage in these cases is also important because of their wide applications.

In this paper, we propose an auto calibration method for high-precision continuous comparators, which can auto calibrate the offset voltage to less than 1 mV when the chip is powered on. A set of correction MOSFETs are added parallel to the main input pair providing offset calibration capability. The calibration procedure is carried out differently on the basis of the first output of the short-input format CMP with a proper common voltage.

This paper is organized as follows. Section 2 introduces the related work, Sect. 3 shows the basic theory and definitions, Sect. 4 depicts our proposed auto offset calibration method in detail, Sect. 5 is performance analysis, and Sect. 6 gives the conclusion and future work.

2. Related Work

There has been much research on high-precision dynamic comparators applied in ADCs,[20–25] but less research on offset calibration for static continuous-time analog comparators. Comparators in ADCs often use switched capacitors to compensate the offset voltage, where clock and sample capacitors are naturally used in ADCs.[20–29] Different clock phases can be used to compensate the comparator offset in normal operation. The sampling capability inherent in ADCs indicates that offset voltage can be periodically sensed, stored, and subtracted in different phases. Different techniques for reducing the offset, such as the optimization of the overdrive voltage of the input transistor[30] and time-domain bulk-driven offset cancellation,[31] have been developed. In time-domain bulk-driven offset cancellation, a technique of trimming offset calibration by modifying the bulk voltage of the comparator input differential pair is proposed. This technique, however, requires the offset voltage to be limited within a certain range to avoid turning on the bulk junctions and deteriorating the performance. A 9-bit *R-2R* resistor array and a 6-bit PMOS junction capacitance array are used for the *N*- and *L-term* compensations,[32] which result in a large area penalty and a limited speed.

In static continuous analog comparator fields, clocks may not be obtainable in some applications for continuous detection. Thus, a universal method should be considered in these

cases. As compared with bipolar transistor counterparts, there is a basic threshold mismatch component present in MOS devices, which results in random offset in the modern standard CMOS process with an order of magnitude larger than the differential-pair offset in bipolar technology.[18] With the bipolar transistors in the Bi-CMOS process, *NPN* differential pairs are used to construct the first two pre-amplifiers to form a three-stage continuous comparator to achieve low offset and good performance simultaneously.[33] Although good specifications are achieved, the Bi-CMOS process is much more complicated and expensive than the standard CMOS process, which considerably increases the chip costs. For a cost-effective design, the standard CMOS process will be preferable.

System offset is commonly generated at circuit design. When different stages are connected to form a multi-stage CMP, the unmatched DC operating points will generate input offset voltage. As the $g_m r_o$ intrinsic gain of MOSFETs is relatively low, the following stages of the CMP will contribute to the input offset voltage, which makes the calculation of the total input offset voltage more complicated. While increasing the gain of the CMP has the benefit of reducing system offset, [34] *cascode* and other gain-boost methods used to increase the gain of the first stage may not be proper methods, owing to the low power supply voltage in advanced technology. On the other hand, power consumption is critical in most handheld or battery-powered devices. Thus, maintaining low-power operation or not clearly increasing the current consumption to lower the offset voltage will be preferable. Circuit techniques can be used to optimize the offset voltage at the cost of a more complicated design, more area penalty, and current consumption, but eliminating the offset voltage is difficult. In modern advanced CMOS processes, where channel length is reduced for higher speed and power supply voltage is lowered for lower power consumption, the intrinsic gain ($g_m r_o$) of MOSFETs further decreases. This poses challenges in increasing the gain of the first stage. Thus, the effect of the following stages of the comparator on the input offset voltage is becoming clearer. Instead of spending a lot of effort to study and reduce the contribution of the offset from each stage, considering the entire comparator as a black box and calculating the total input offset voltage without delving into the internal circuit details are straightforward tasks to accomplish at the system's top level. By this method, the comparator offset voltages of different structures and designs, and even in different processes can be calibrated according to the same method, which makes it universal to general-purpose comparators.

According to the above analysis, a simple and universal method without increasing bias current is proposed to adapt new applications. In the fields of continuous detection applications, designing a comparator with auto calibration capability in a state-of-the-art process will be cost-effective. Since in the wafer manufacturing process, each die will suffer from a different corner environment, even the post-package process can generate stress affecting the device parameters. If the calibration is carried out at each chip in the *CP* and *FT* test, the extra test cost will be clear. Therefore, employing an auto calibration feature in the comparator during each power-up ensures that these factors have minimal impact on normal operation and helps minimize testing costs. Furthermore, the proposed calibration method can be easily extended to reach the expected specification and shifted to other advanced processes, which make it a simple and promising method.

3. Basic Theory Model and Definitions

Table 1 gives the symbol representation table to be used in this section. Figure 1 is a commonly used N-channel metal-oxide-semiconductor (NMOS) input comparator, and M_0/M_1 forms an input pair. If the effect of the compensation NMOS M_n is not considered first, assuming that the offset voltage of the comparator is V_{OS}, the current flowing through M_0 will be greater than M_1 and the comparator will flip in advance. To counteract the effects of V_{OS}, which means that the comparator flips exactly at $V_A = V_B$, a set of calibration NMOS M_n is added; thus, in the equilibrium state, the current flowing through the left branch is equal to that flowing through the right branch, which is equal to I_{SS}/2. Thus,

$$\frac{1}{2}\mu_n C_{ox}\left(m\frac{W}{L}\right)\left(V_{GS}+V_{OS}-V_{TH}\right)=\frac{1}{2}\mu_n C_{ox}\left(m+n\right)\frac{W}{L}\left(V_{GS}-V_{TH}\right)^2. \tag{1}$$

From Eq. (1), we obtain Eqs. (2) and (3).

$$\sqrt{m}\left(V_{GS}+V_{OS}-V_{TH}\right)=\sqrt{m+n}\left(V_{GS}-V_{TH}\right) \tag{2}$$

$$V_{OS}=\left(\sqrt{1+\frac{n}{m}}-1\right)\left(V_{GS}-V_{TH}\right) \tag{3}$$

Table 1
Symbol representation table.

Symbol	Representation
V_{OS}	Offset voltage
W/L	Size ratio
I_{SS}	Bias current
μ_n	Electron mobility
C_{ox}	Unit gate capacitance
V_{GS}	Gate-source voltage
V_{TH}	Threshold voltage

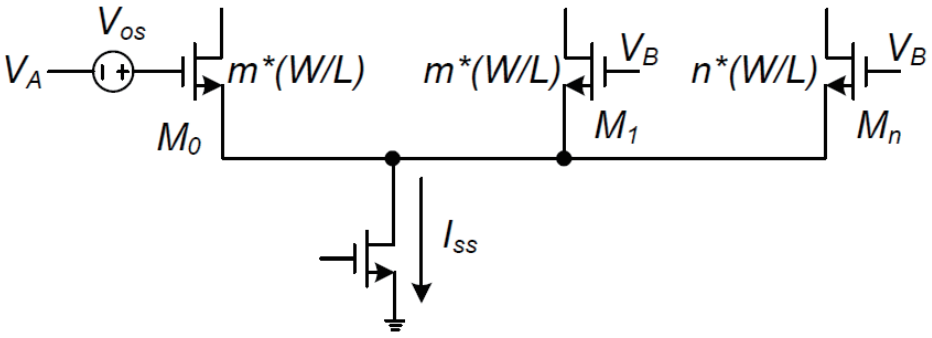


Fig. 1. Comparator input pair and calibration NMOS M_n.

From Eq. (3), V_{OS} and n are in square root relationships. Then,

$$\frac{\partial V_{OS}}{\partial n} \propto \frac{1}{\sqrt{n}}. \tag{4}$$

We obtain the maximum calibration step at $n = 1$, which is

$$V_{step,\max} = V_{OS,n=1} = \left(\sqrt{1+\frac{1}{m}} - 1\right)\left(V_{GS} - V_{TH}\right). \tag{5}$$

Moreover, in the equilibrium state, we have

$$\frac{I_{SS}}{2} = \frac{1}{2}\mu_n C_{ox}\left(m+n\right)\frac{W}{L}\left(V_{GS} - V_{TH}\right)^2. \tag{6}$$

Thus,

$$V_{GS} - V_{TH} = \sqrt{\frac{I_{SS}}{\mu_n C_{ox}\left(m+n\right)\frac{W}{L}}}. \tag{7}$$

Putting Eq. (7) in Eq. (5), we obtain

$$V_{step,\max} = V_{OS,n=1} = \left(\sqrt{1+\frac{1}{m}} - 1\right)\sqrt{\frac{I_{SS}}{\mu_n C_{ox}\left(m+1\right)\frac{W}{L}}}. \tag{8}$$

As shown in Eq. (8), the adjusted step is inversely proportional to the square root of W/L, inversely proportional to m, and proportional to the square root of I_{SS}. The larger the m, the smaller the $V_{step,\max}$.[34] By selecting the proper weights of M_n to the input pair, we can obtain the required calibration accuracy.

4. Proposed Auto Offset Calibration Method

On the basis of the theoretical analysis in the previous section, we propose in this paper an auto calibration method. When the chip is powered on, the digital control logic is used to search for a suitable calibration NMOS size on the comparator input pair, so that the comparator flips exactly at $V_A = V_B$. That is, the offset voltage of the comparator is counteracted by introducing a suitable calibration NMOS size. Calibration flow is as follows: V_A and V_B are shorted and biased

by a suitable common-mode voltage (to ensure that the comparator is within the common-mode voltage range of normal operation). At the same time, the digital control circuit searches the trim bits from low to high, and then from high to low, the calibration code at which the comparator first switches state is determined sequentially. These two calibration codes are added and averaged to obtain the actual code. This mainly takes into account the fact that there may be a certain delay in comparator flipping, which leads to deviations from the optimal calibration code. This part of the search error can be counteracted by averaging the results of searching from low to high and from high to low. The specific calibration flowchart is shown in Fig. 2. Two calibration routines are selected differently on the basis of the CMP's initial output, which decides the *left<3:0>* or *right<3:0>* code to be scanned. The *Verilog-HDL* code realization of the calibration scan logic is given in Appendix at the end of this paper.

As shown in Fig. 3, circuit implementation contains two parts. One consists of the comparator core and calibration circuit, and the other is the calibration control logic. Two NMOSs with M = 32 constitute the main part of the input pair, whereas the left and right part calibration NMOSs

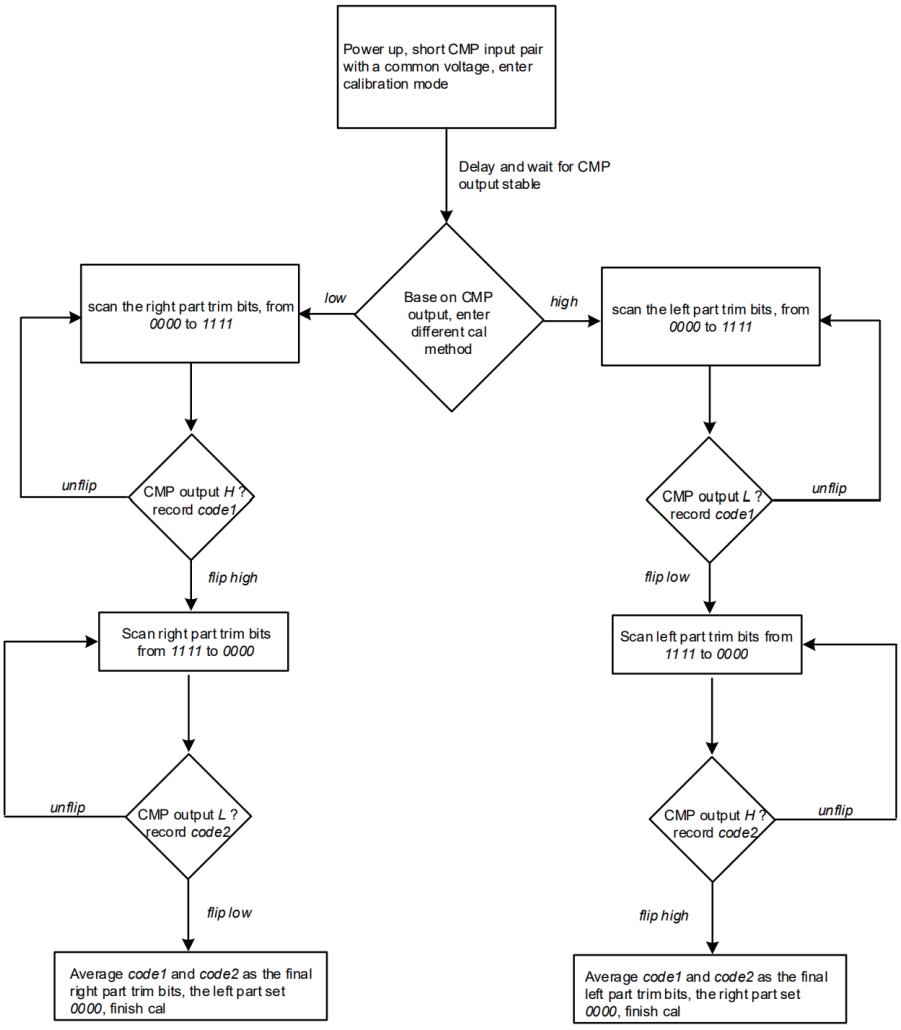


Fig. 2. Process of auto calibration.

are weights of 1/2/4/8. According to different design needs, different trim step sizes can be obtained by adjusting the relative ratio of the smallest calibration NMOS size to the main input pair. *MP3/MP4* form local positive feedback to increase gain. The output is shaped by two stages of *INV*, then it is directed to the digital control circuit. *Left<3:0>* and *right<3:0>* trim bits are controlled by the calibration logic. Once the power-on calibration is completed, these values are latched at the final calibration value.

As an example, let us assume that a positive offset voltage V_{OS} exists at *INN*. This means that if we short *INN* and *INP*, the bias current is predominantly directed through the *INN* input pair, resulting in a CMP output, that is lower than that in the case of the *INP* input pair. According to the calibration routine, *left<3:0>* will be set to *0000*, and calibration will start by scanning *right<3:0>* from *0000* towards *1111*. The control logic monitors the CMP output and records the first trim bit value *A* at which it flips highly. Then, *right<3:0>* is set to *1111*, so the CMP output remains high. The control logic scans *right<3:0>* from *1111* towards *0000* and records the second trim bit value *B* at which it becomes low. A simple average algorithm is used to obtain the final trim value. Thus, *right<3:0>* has the trim value and *left<3:0>* remains *0000*. Then, the procedure is finished and the CMP can perform normal operation.

5. Performance Analysis

Circuit design is based on the 110 nm flash process with the 5 V IO device. The calibration control logic is implemented in a digital manner. As shown in Fig. 4, the digital calibration control logic contains an enable signal, which is related to the chip *POR* signal and a clock input to make the digital logic work normally during calibration. However, when the process is finished, all the trim bits are latched to their final value and *clkin* is blocked. As a result, the digital calibration logic remains static and does not generate any static current during the normal operation of the comparator. The *OUT* signal serves as feedback from the comparator to the control logic, enabling it to determine the initial output state of the comparator and the trim bits

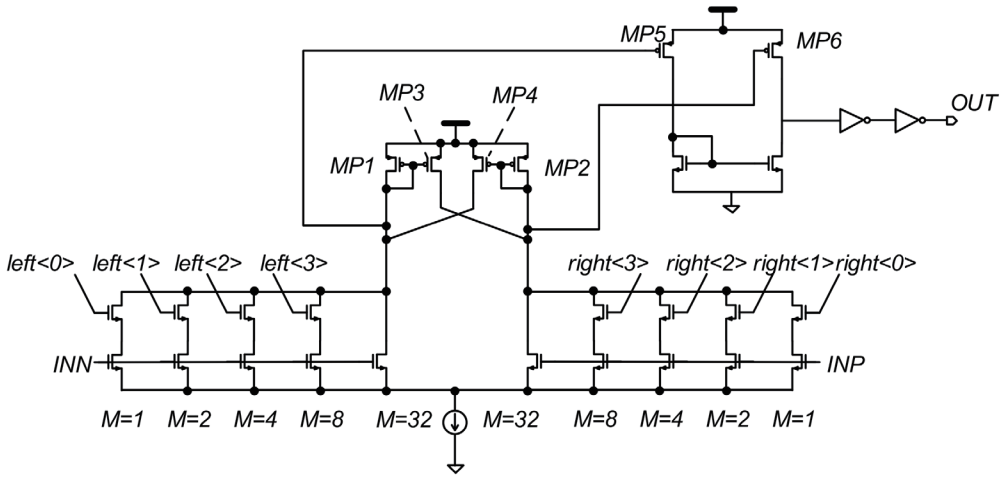


Fig. 3. Circuit implementation of the comparator and calibration NMOS.

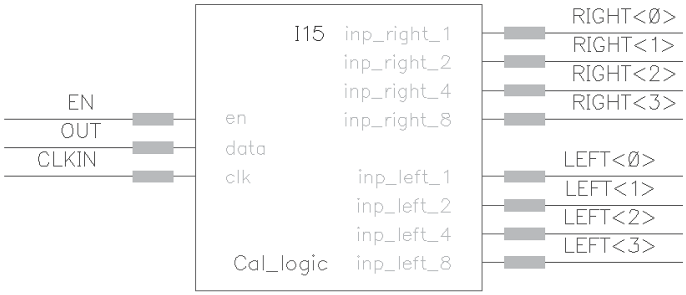


Fig. 4. Digital calibration control logic block.

at which it flips.

To verify the validity of the calibration, 5 and 8 mV offset voltages are added artificially at the *INN* end to determine whether the calibration procedure can eliminate the offset. *Mix-sim* is performed on the *VCS/XA* platform with a 32 KHz low-speed input clock. Choosing 32 KHz as the input *clk* is based on the fact that the general-purpose Microcontroller Unit (MCU) commonly integrates a low-speed clock for watching dog timing. Thus, in hardware, the calibration scheme only adds some calibration NMOS and a small part of the digital control logic without increasing current consumption, which can be easily shifted to other advanced processes. As shown in Fig. 5, after the *EN* signal is enabled, the control logic detects that the comparator output is equal to 0, so the trim bits start to scan from *0000* to *1111* at the right end, and the comparator flips at *0101*. Then, the trim bits are forced to *1111* and start to scan down from *1111*, and the comparator flips at *0100*. The average output of *0100* is to *right<3:0>* and keeps *left<3:0>* low. As shown in Fig. 5, the digital logic flips the trim bits at the rising edge of *clkin* and detects the comparator output at the falling edge of the clock denoted by *T1/T2*, respectively.

As in the case of the 8 mV offset, as shown in Fig. 6, the comparator output flips at *1000*, then the trim bits are forced to *1111* and scan down to *0111* at which the CMP flips down. Average trim bits *0111* are given to *right<3:0>*. Figure 7 shows the process of verifying the calibration effectiveness by transient simulation after obtaining the trim bits by the above method. Given the *INP* 2.5 V and *INN* 5 mV offset voltages, *INN* scans upwards from below *INP* and flips at 2.4996 V [Fig. 7(a)]. Then, *INN* scans downwards from above *INP* and flips at 2.49915 V [Fig. 7(b)]. This indicates that the calibrated comparator offset voltage is within 1 mV. Figure 8 shows the case of *INN* with 8 mV offset voltage; *INN* scans upwards from below *INP* and flips at 2.49973V [Fig. 8(a)]. Then, *INN* scans downwards from above *INP* and flips at 2.49928V [Fig. 8(b)]. Again, it shows that the calibrated comparator offset voltage is within 1 mV. Figures 9 and 10 are the cases of adding offset to the *INP* side. Similarly to the case of the *INN* side, the control logic scans from *0000* to *1111*, then from *1111* to *0000*. The sole distinction lies in the fact that the initial output of the CMP determines which side of the trim bits requires scanning. As shown in Figs. 9 and 10, the final values are the same as those in Figs. 5 and 6, which proves the functionality of the control logic. Figure 11 shows another calibration process without adding offset. The CMP output first flips at *0001*, then flips at *0000* when scanning down. The final

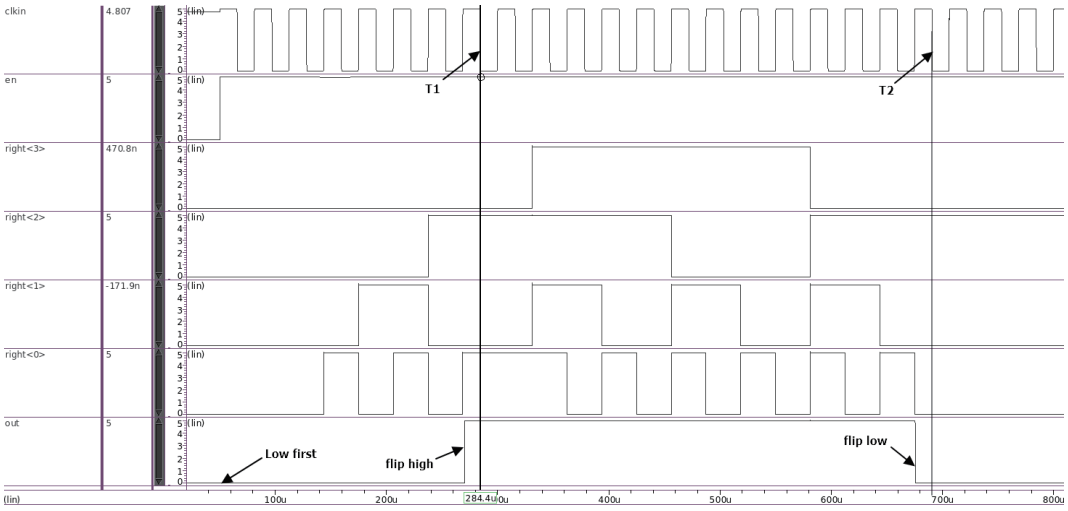


Fig. 5. (Color online) Calibration process for *INN* with 5 mV offset.

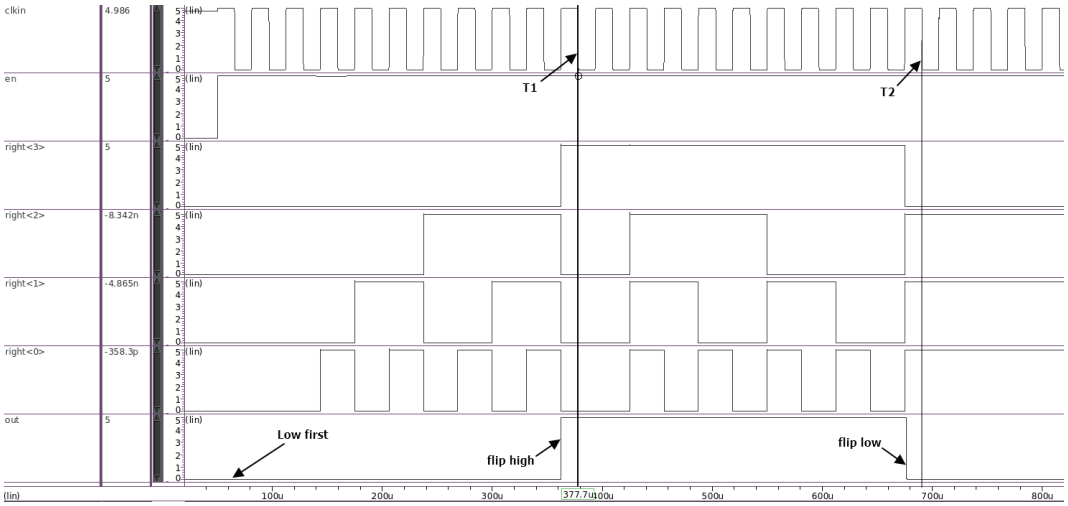
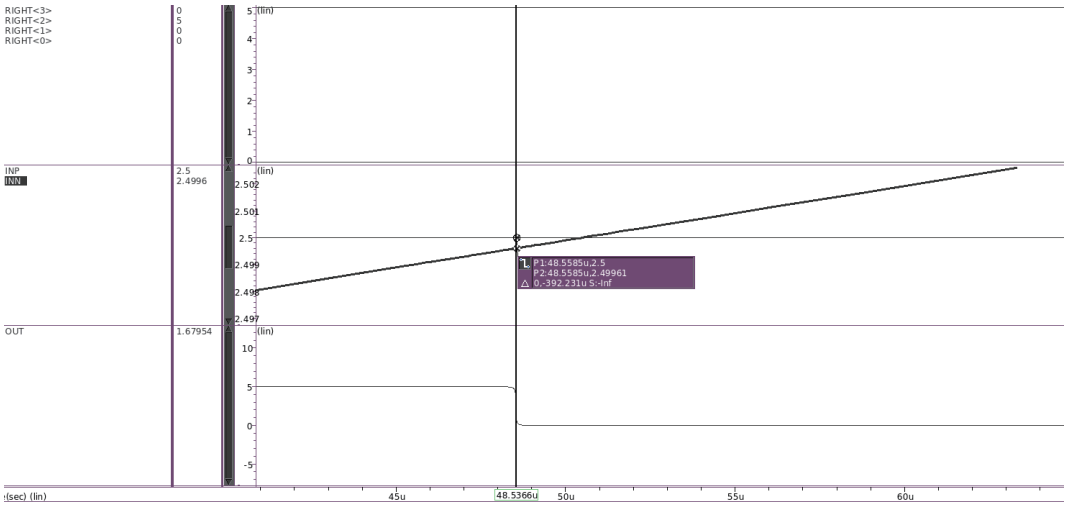


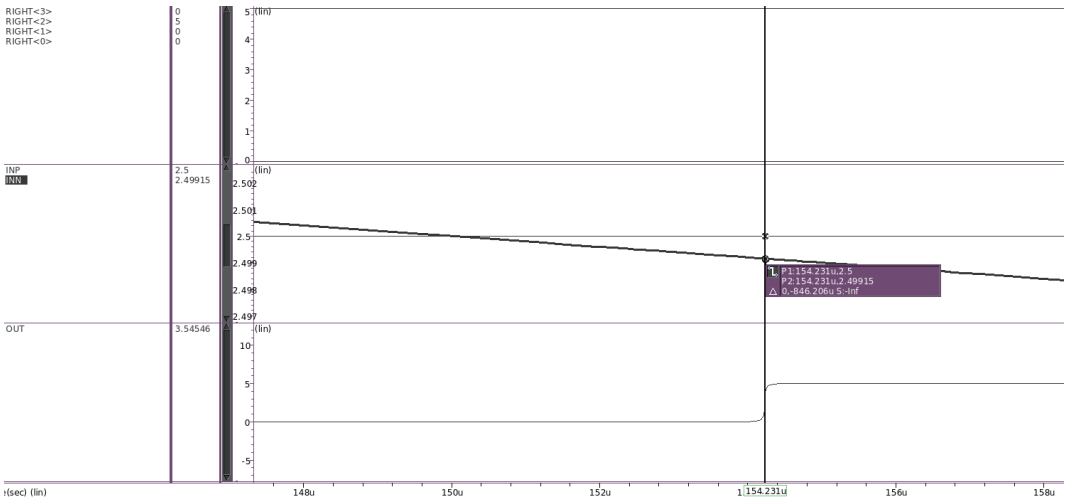
Fig. 6. (Color online) Calibration process for *INN* with 8 mV offset.

average result is that both *left<3:0>* and *right<3:0>* are *zero*, indicating that the calibration process does not introduce additional offset. Since the power-on process usually lasts for several milliseconds before entering normal operation, the calibration time is acceptable in general applications.

The simulations primarily concentrate on the 5 mV offset as it aligns with the typical range of commercial-use comparators. It can be easily concluded that a larger trim range can be made by adding more trim bits and that trim steps can be made smaller by making *m* larger. Both are easily realized in system design.

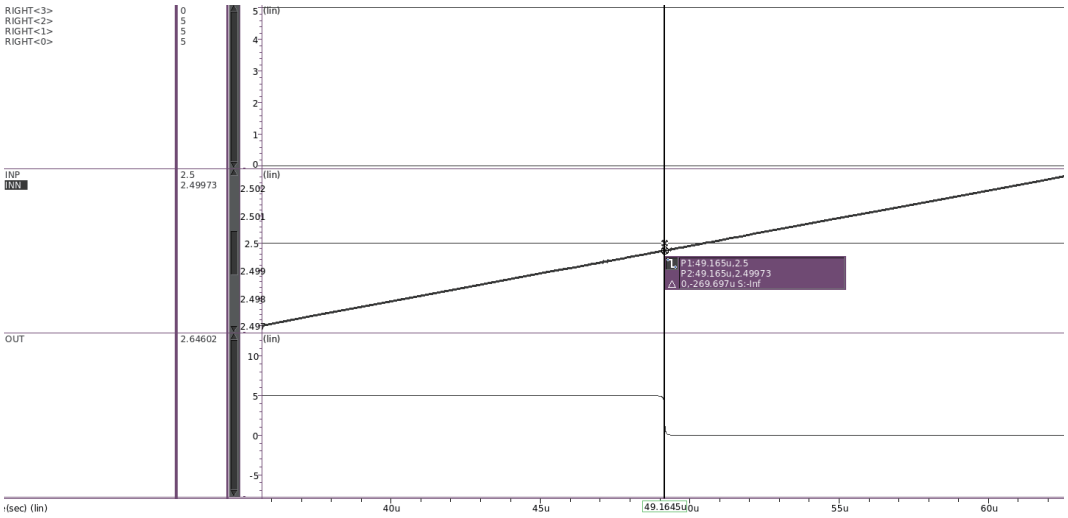


(a)



(b)

Fig. 7. (Color online) Transient simulation to verify the calibration effect with *INN* and 5 mV offset, voltage *right<3:0>=0100*. (a) *INN* scans upwards from 2.495 to 2.505 V after calibration. (b) *INN* scans downwards from 2.505 to 2.495 V after calibration.



(a)

(b)

Fig. 8. (Color online) Transient simulation to verify the calibration effect with *INN* and 8 mV offset voltage, *right<3:0>=0111*. (a) *INN* scans upwards from 2.495 to 2.505 V after calibration. (b) *INN* scans downwards from 2.505 to 2.495 V after calibration.

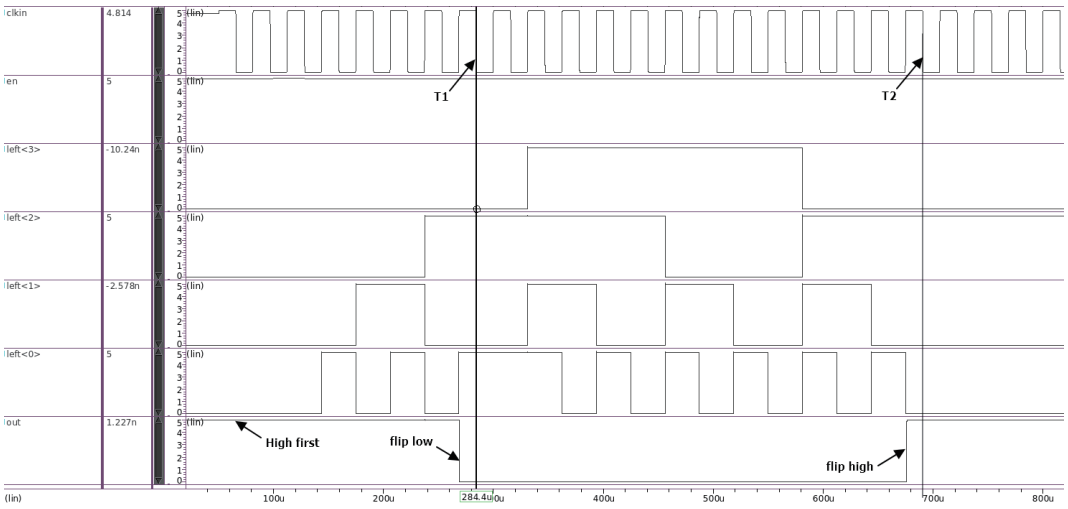


Fig. 9. (Color online) Calibration process for *INP* with 5 mV offset voltage.

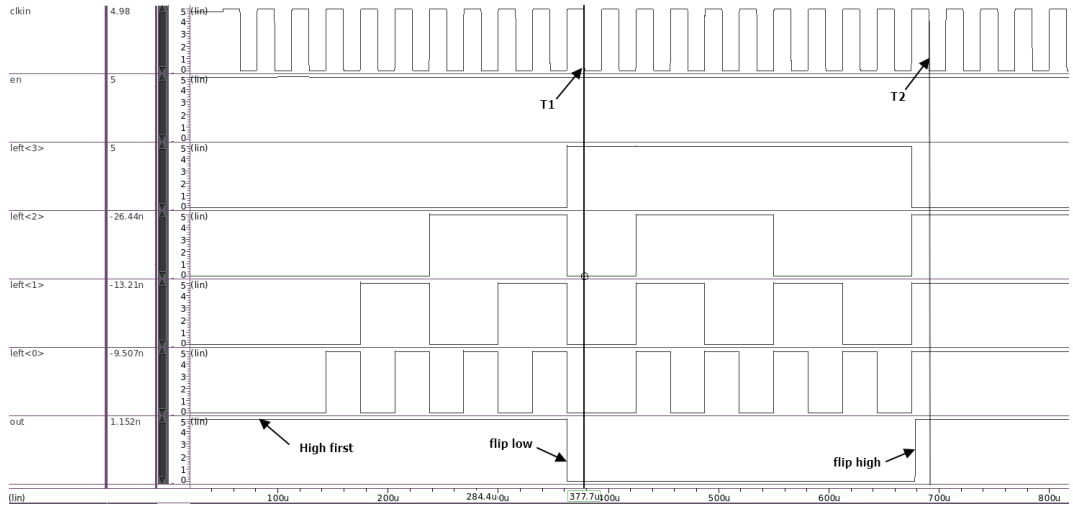


Fig. 10. (Color online) Calibration process for *INP* with 8 mV offset voltage.

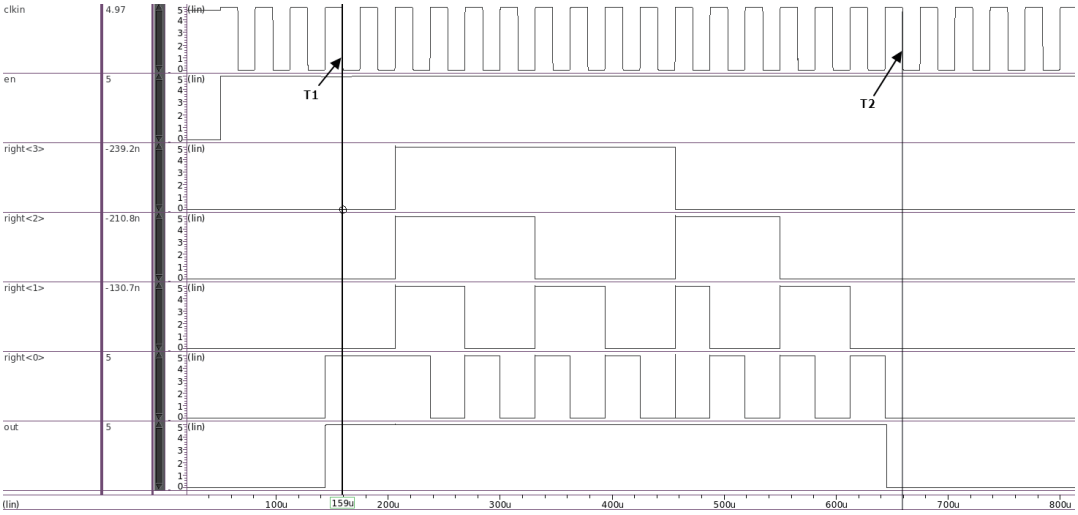


Fig. 11. (Color online) Calibration process of adding zero offset voltage.

6. Conclusion and Future Work

In this paper, we proposed an auto calibration method for continuous comparators after power-on. Combined with theoretical analysis and simulation verification, this method can auto search for the best calibration value on the basis of the comparator output. During the design stage, the calibration error can be fine-tuned by selecting distinct weights that are proportional to the NMOS size, aligning with the specific requirements of the application. Transient simulation verifies the functionality and reliability of the scheme. The proposed auto calibration method does not increase the current consumption and can be easily shifted to other advanced technology processes.

For future use, the current scheme is based on power-on calibration. Considering environmental variations, the calibration procedure can be run at any time as required by the top-level system. On the other hand, calibration is carried out at a fixed common mode voltage, but it would be better if it can be done at a common mode voltage according to actual applications.

Appendix

```
module cmp_calibrate
(en, data, clk, inp_right_1, inp_right_2, inp_right_4, inp_right_8, inp_left_1, inp_left_2, inp_left_4, inp_left_8);
input en, data, clk;
output inp_right_1, inp_right_2, inp_right_4, inp_right_8, inp_left_1, inp_left_2, inp_left_4, inp_left_8;
reg [3:0] inp_right=0;
reg [3:0] inp_left=0;
reg [3:0] sub_count=4'b1111;
reg [3:0] acc_count=0;
reg [3:0] average=0;
reg refer=0;
reg refer2=0;
reg flag=0;
reg acc_sw=1;
reg sub_sw=1;
reg scan_en=1;
reg [4:0] i=0;
reg [4:0] j=0;
reg [3:0] sub_final=0;
reg [3:0] acc_final=0;
assign {inp_right_8,inp_right_4,inp_right_2,inp_right_1}=inp_right;
assign {inp_left_8,inp_left_4,inp_left_2,inp_left_1}=inp_left;
always @(posedge clk)
begin
  if(en)
  begin
    if(!flag)
      begin
        refer=data;
          refer2=~data;
        flag=1;
          acc_sw=0;
          sub_sw=1;
          sub_count=4'b1111;
          acc_count=0;
      end
    else
      begin
        if(refer)
          begin
            if(scan_en)
            begin
              if(!acc_sw)
                begin
            if(data==refer)
              begin
                  inp_left=acc_count;
                  inp_right=0;
```

```
    j=j+1;
  acc_count=acc_count+1;
  if(j==17)
  begin
        $display("--------------------------------------");
    $display("the data from 0 to F does not change");
    scan_en=0;
        inp_left=4'b1111;
        inp_right=0;
    j=0;
      average=4'b1111;
    end
        end
  else
  begin
          acc_final=acc_count-1;
          $display("--------------------------------------");
    $display("the data changed at the acc_count:%d",acc_final);
          acc_sw=1'b1;
          sub_sw=1'b0;
  end
end
  else if(!sub_sw)
    begin
    if(data==refer2)
      begin
        inp_left=sub_count;
        inp_right=0;
      sub_count=sub_count-1;
            i=i+1;
      if(i==17)
      begin
        $display("the data from F to 0 does not change");
            sub_count=0;
            i=0;
            sub_sw=1;
        scan_en=0;
        average=4'b1111;
      end
          end
    else
      begin
          sub_final=sub_count+1;
          average=(acc_final + sub_final)/2;
      $display("the data changed at the sub_count:%d",sub_final);
          $display("average: %d",average);
      $display("--------------------------------------");
          sub_sw=1'b1;
          scan_en=1'b0;
    end
  end
    end
    else
    begin
      inp_left=average;
      inp_right=0;
    end
```

```
        end
    else
      begin
        if(scan_en)
        begin
          if(!acc_sw)
            begin
        if(data==refer)
          begin
            inp_right=acc_count;
            inp_left=0;
        j=j+1;
          acc_count=acc_count+1;
            if(j==17)
            begin
          $display("---------------------------------------");
          $display("the data from 0 to F does not change");
                scan_en=0;
                inp_right=4'b1111;
              inp_left=0;
                j=0;
                average=4'b1111;
          end
            end
        else
        begin
                acc_final=acc_count-1;
            $display("---------------------------------------");
            $display("the data changed at the acc_count:%d",acc_final);
                acc_sw=1'b1;
                sub_sw=1'b0;
          end
      end
            else if(!sub_sw)
              begin
            if(data==refer2)
              begin
                  inp_right=sub_count;
              inp_left=0;
                sub_count=sub_count-1;
                  i=i+1;
                if(i==17)
                begin
                  $display("the data from F to 0 does not change");
                        sub_count=0;
                        i=0;
                        sub_sw=1;
                  scan_en=0;
                  average=4'b1111;
              end
                end
        else
          begin
      sub_final=sub_count+1;
      average=(acc_final + sub_final)/2;
$display("the data changed at the sub_count:%d",sub_final);
      $display("average: %d",average);
```

```
            $display("-------------------------------------");
              sub_sw=1'b1;
              scan_en=1'b0;
          end
          end
            end
            else
            begin
              inp_right=average;
              inp_left=0;
            end
          end
          end
    end
end
endmodule
```

References

1 C. Lin, Y. He, and N. Xiong: 5th Int. Symp. Parallel and Distributed Computing (IEEE, 2006) 148.
2 F. Xia, R. Hao, and J. Li: J. Systems Architecture **59** (2013) 1231.
3 P. Kumar, R. Kumar, and G. Srivastava: IEEE Trans. Network Sci. Eng. **8** (2021) 2326.
4 C. Wu, C. Luo, and N. Xiong: IEEE Access **6** (2018) 20021.
5 H. Cheng, Z. Xie, and Y. Shi: IEEE Access **7** (2019) 117883.
6 Y. Yao, N. Xiong, and J. Liu: Comput. Math. Appl. **65** (2013) 1318.
7 S. D'Amico, G. Cocciolo, and A. Spagnolo: IEEE Trans. Instrumentation and Measurement **63** (2014) 295.
8 T. Forzley and R. Mason: Integrated Circuits and Systems Design (SBCCI, 2013) 1.
9 J. Li, G. Wu, and J. Wu: IEEE Open J. Circuits Syst. (2007) 28.
10 V. Varshney, A. Kumar, and A. K. Dubey: Int. Conf. Electrical, Electronics and Computer Engineering (UPCON, 2019) 1.
11 A. A. Basem and N. M. Ahmed: AEU Int. J. Electron. Commun. **146** (2022) 154116.
12 Y. Guo, T. Lu, and Z. Wang: Microelectronics Computer **28** (2011) 50.
13 B. Razavi and B. A. Wooley: IEEE J. Solid-State Circuits **27** (1992) 1916.
14 H. Jeon and Y. Kim: IEEE Int. SOC Conf. (2010) 285.
15 A. Gupta, A. Singh, and A. Agarwal: AEU Int. J. Electron. Commun. **134** (2021) 153682.
16 H. A. Majid and Y. Yusoff: IEEE Int. Circuits and Systems Symp. (ICSyS, 2015) 40.
17 J. Yang, X. Wang, and L. Li: J. Semicond **32** (2011) 93.
18 B. Razavi: Design of Analog CMOS Integrated Circuits (McGrawx-Hill Education. 2017).
19 PIC16F684 datasheet: http://www.microchip.com/en-us/produce/PIC16F684 (accessed July 2023).
20 J. Li, W. Xu, and Y. Yu: J. Semicond. **31** (2010) 80.
21 H. yousefi and S. M. Mirsanei: Iranian Conf. Electrical Engineering (ICEE, 2020).
22 M. M. Hossain and S. N. Biswas: Int. Conf. Advances in Electrical Engineering (ICAEE, 2019).
23 J. He, S. Zhan, and D. Chen: IEEE Trans. Circuit and Systems I: Regular Papers **56** (2009) 911.
24 S. Babayan-Mashhadi and R. Lotfi: IEEE Trans. Very Large Scale Integ. Systs. **22** (2014) 343.
25 A. Khorami, R. Saeidi, and M. Sachdev: Microelectron. J. **102** (2020) 104842.
26 X. Tong, Z. Zhu, and Y. Yang: J. Semicond. **33** (2012) 15011.
27 V. Deepika and S. Singh: Int. Conf. Nanomaterials and Technologies (CNT-2014) Prodedia Materials Science **10** (2015) 314.
28 Q. He, X. Zhou, and Q. Li: IEEE Int. Conf. ASIC (ASICON, 2017).
29 G. Sun, Z. Yin, and L. He: IEEE Int. Conf. Electron Devices and Solid-State Circuits (2014).
30 M. Miyahara and A. Matsuzawa: IEEE Asian Solid-State Circuits Conf. (2009) 233.
31 J. Lu and J. Holleman: IEEE Trans. Circuits and Systems I: Regular Papers **60** (2013) 1158.
32 R. Y. Choi and C. Tsui: Int. Midwest Symp. Circuits and Systems (MWSCAS, 2012).
33 J. Yang and Q. Y. Feng: Int. Conf. Intelligent Human-Machine Systems and Cybernetics (2013) 466.
34 P. R. Gray, P. J. Hurst, and Lewis: Analysis and Design of Analog Integrated Circuits, (Wiley, New York, 2001) 4th ed.