A DIGITAL DESIGN FLOW FOR MOS CURRENT-MODE LOGIC

Undergraduate Research Final Report - High-Efficiency Circuit and System Lab

Department of Engineering Science and Ocean Engineering, National Taiwan University
Jing-Hua Chang, Yun-Chung Lee, Jau-Horng Chen

ABSTRACT

In this report, a digital design flow for implementing MCML circuits is presented. MCML circuits are fully differential and are not directly supported by commercial CAD tools. A special design methodology must be developed such that design of MCML circuits can be automated using standard CAD tools. MCML standard cells are first designed and the cell libraries required for commercial CAD tools are then generated. Synthesis and place & route (P&R) in a single-ended domain that tricks commercial CAD tools into treating MCML circuits as standard CMOS logic circuits. An algorithm that automatically converts single-ended circuits into fully differential MCML circuits is developed. To achieve the theoretical high-frequency operation of the MCML circuits, I devised two methods to enhance the MCML library to speed up the overall performance.

1. INTRODUCTION

MOS current mode logic (MCML) is an alternative to conventional CMOS logic for high-speed logic design that is widely used in high-speed wireline communication. The working principle of the MCML circuit is to redirect the current of a constant current source through a fully differential network of input transistors and use the differential voltage drop of a pair of complementary load devices as the output. Since the transistors operate constantly in the saturation region, switching noise in the supply rail is minimized, which has gained interests lately in encryption circuits.

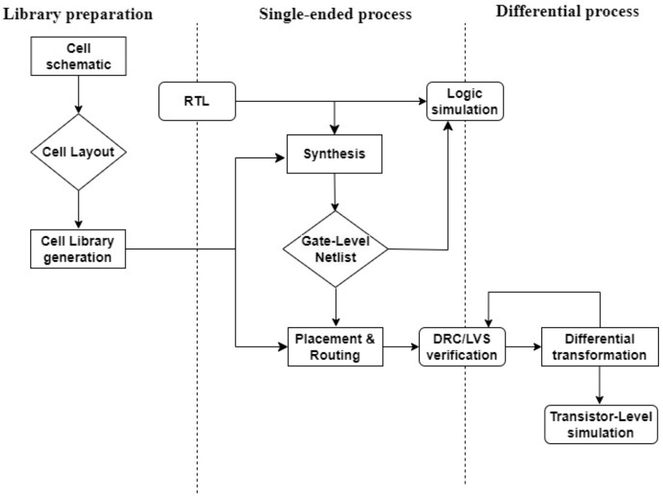
Design automation for conventional CMOS logic is very mature. However, commercial design automation tools for MCML circuits are virtually non-existent. In [1], design automation using conventional CMOS logic design tools were proposed. The main benefit of using proven conventional tools is the algorithm's correctness. Instead of reinventing new CAD tools for MCML circuits, only tools for translating between MCML and CMOS logic circuits are needed. The goal is basically fooling conventional CMOS tools to process MCML circuits. In [1], the integration of conventional CAD tools and homemade tools was focused while chips were not actually implemented. In this work, missing pieces were filled to actually tape out a real chip.

2. PREVIOUS WORK

The main difference between CMOS logic and MCML for commercial CAD tools is single-ended I/O and fully differential I/O. Starting with the library characterization, commercial tools such as Synopsys SiliconSmart and Cadence Liberate supports only single ended logic cells. Synthesis tools such as Synopsys Design Compiler and Cadence GENUS support logic cells that comply with certain template, which are single ended as well. Place and route tools such as Synopsys IC Compiler and Cadence Innovus can take fully differential netlists and treat all wires as single-ended, but the benefits of being differential in MCML circuits will be degraded. In this work where CMOS logic are replaced with MCML, three main steps are used namely library preparation, single-ended process, and differential process. The design flow is illustrated in Fig. 1.

Fig.1. Design flow

2.1. Library preparation



The basis of digital implementation is the standard cell library. For MCML circuits, the standard cell library is not readily available and must be implemented. The final performance of the circuit designs will significantly depend on the cell library. Before synthesis, the standard cells need to be designed, then each cell has to be characterized, finally a timing library for synthesis tools needs to be generated.

2.1.1. *Standard cell schematic design and layout*

The cells utilized in this work were based on the schematic design shown in [1] and [2]. The logic cells were targeted for UMC's 0.18um CMOS process with 6 metal layers. For the logic cells, only metal layer 1 was utilized such that metal layers 2 to 5 can be used for routing. The cells created during the library generation process for MCML in this work are shown in the Table.1 below,

Gate	Drive strengths
INV	1 , 2 , 4 , 8
BUF	1 , 2 , 4 , 8
AND	1
NAND	1
OR	1
NOR	1
XOR	1
XNOR	1
DFF	1
DL	1

Table.1 standard cell list

2.1.2. *Library Generation*

First, we need to generate a timing library for the synthesis tool and place and route tools (P&R) from the extracted netlist, which is shown in Fig.2. Additionally, a library exchange format (LEF) file from the layout has to be generated for the P&R tool, which provides pin locations and blocking metal layers for performing routing between cells. With these libraries, synthesis and P&R can then be proceeded. While the design of MCML cells were performed differentially, single-ended libraries have to be generated to trick the commercial design tools.

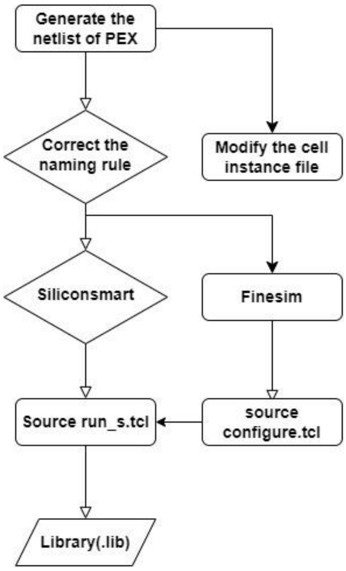


Fig.2. Timing library generation flow

2.2. Single-ended process

For commercial CAD tools, only conventional CMOS logic cells with single-ended pins are supported as described in [3]. The goal of this work is to utilize conventional digital design flow as much as possible. The single-ended process is basically the same as the conventional CMOS digital design flow as described in [4]. First, the Verilog RTL code is synthesized with Synopsys Design Compiler, which creates the gate-level netlist or synthesized Verilog file. Next, P&R of the previous gate-level netlist is performed with Cadence Innovus. Preliminary DRC and LVS can be performed inside Innovus. However, the cells used inside Innovus are not true MCML cells, but single ended cells used to trick Innovus. Moreover, routing between cells are single-ended fat wires as described in [1]. Finally, the single-ended layout is exported as a GDS file, which can then be further processed in Cadence Virtuoso. A single-ended Verilog netlist after routing, which includes additional clock buffers after clock tree synthesis, is also generated for use in Cadence Virtuoso Schematic Entry. After this step, the singled-ended process is concluded with detailed steps of the flow shown in Fig.3.

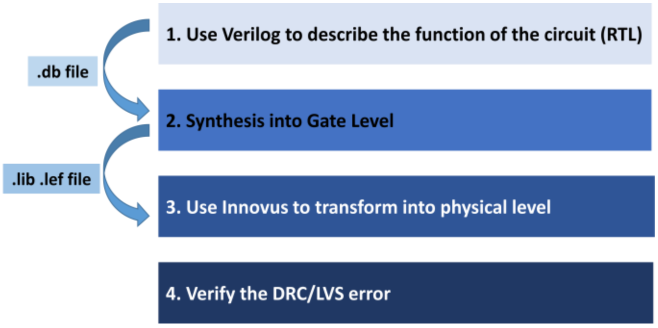

1. Use Verilog to describe the function of the circuit (RTL)
.db file
2. Synthesis into Gate Level
.lib .lef file
3. Use Innovus to transform into physical level
4. Verify the DRC/LVS error

Fig.3. single-ended design flow

2.3. Differential process

The differential process of the MCML flow converts the single ended layout and netlist into differential layout and netlist, which can be further validated with conventional full-custom design flow. The layout GDS file previously generated is imported into Cadence Virtuoso. First the single-ended MCML cells have to be replaced with real differential MCML cells. Then, single-ended routing fat wires have to be replaced with differential wires as described in [5]. Unlike single-ended cells, differential cells that are flipped during the placement step can lead to inverting of routing wires. Therefore, cells must be replaced to ensure logic correctness.

To generate the schematic for Cadence Virtuoso, the single-ended Verilog netlist after routing is imported. Similar to the imported layout, single-ended cells are replaced with differential cells. The flipped cells previously mentioned that are replaced to ensure logic correctness are also replaced in the schematic.

For ease of dealing with the layout and schematic files in Cadence Virtuoso, scripts are written in Cadence SKILL language that are run within the Cadence Virtuoso environment. Theoretically, the matched netlist and layout exported by Cadence Innovus should also be matched after the SKILL script is executed if the algorithm is bug free. DRC and LVS of the differential layout and schematic are performed using Mentor Graphics Calibre. During the development of this design flow, numerous iterations were performed so that the LEF file used to trick Cadence Innovus can lead to a DRC error free differential layout in this step. With the conclusion of the differential process of this flow, layout can then be exported for tape out. The detail steps of algorithm implemented with Cadence SKILL that performs the differential transformation is shown in Fig.4.

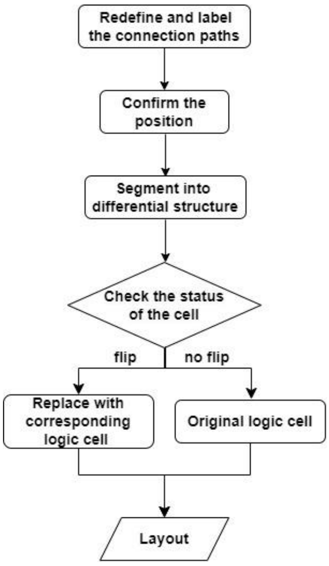

Redefine and label the connection paths
Confirm the position
Segment into differential structure
Check the status of the cell
flip
no flip
Replace with corresponding logic cell
Original logic cell
Layout

Fig.4. Differential Transformation Algorithm

3. DESIGN METHODOLOGY

My contribution to this research is to design 4 different XORs and integrate them into the CML standard cell library. Without XORs in the library, the synthesis tools will use 2 NOT gates, 2 AND gates, and 1 OR gate to represent an XOR gate. The usage of XORs therefore occupy large area. With the well-designed CML XORs added to the library, the overall area reduced by 37% and the power consumption dropped 33% for a sigma-delta modulation DAC.

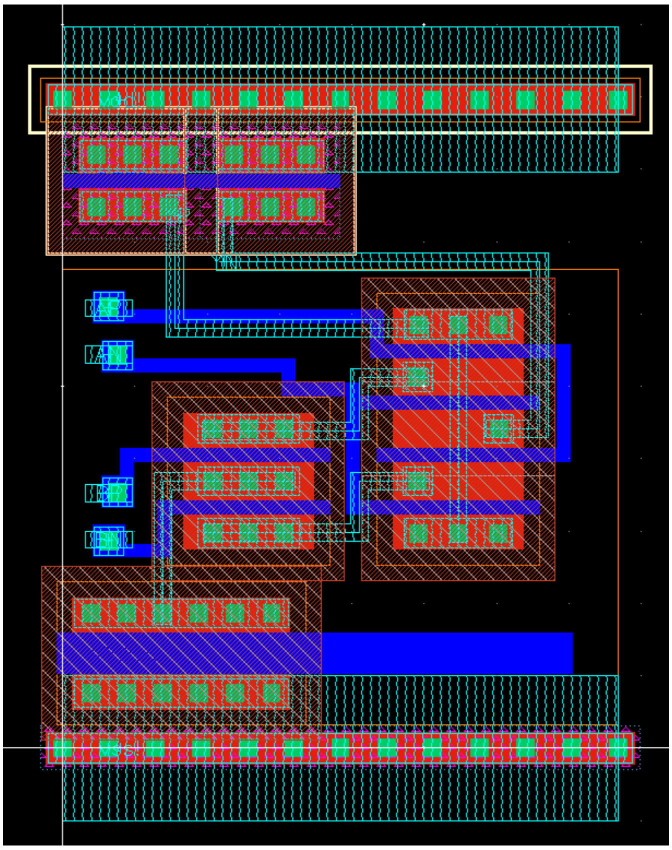


Fig.5. MCML XOR Layout

One of the problems in the taped-out chip is that the DFFs cause great latency. Since DFF is an essential element in digital design, improving its speed became an urgent task. The original MCML DFF cell applied a standard structure. Which is clean enough to be implemented into the digital design flow. However, aiming to achieve theoretical high-speed operation, I looked for new circuit topology of high-frequency MCML DFF.

Three methods are implemented into the high-speed DFF design. I implement the design topology proposed in [6]. We can double the tail current without increasing the size of the transistors, and thus reduce the input capacitance. Additionally, the cross coupled pair and a biasing tail current source can help lower the signal transition times of the latch.

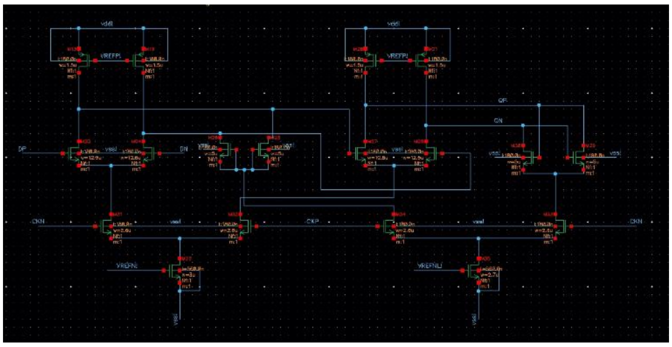


Fig.6. Original Design of D Flip-Flop

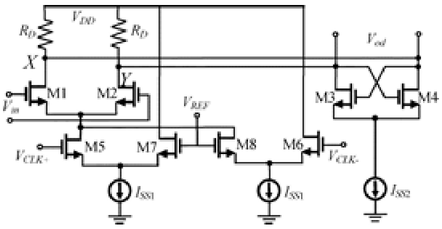


Fig.7. Proposed Novel Structure of High-Speed D Flip-Flop

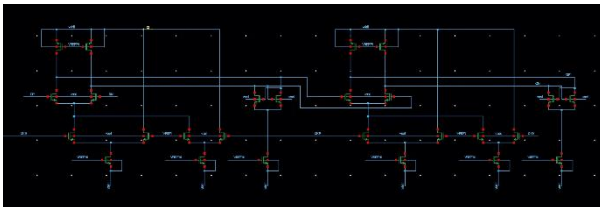


Fig.8. Fine-tuned Novel Structure of High-Speed D Flip-Flop

The layout of the proposed DFF is 3 times larger than the original design. The simulation speed is approximately the same as the original one. The conjecture of the result is that some of the transistors are not operating in the saturation region. In the future work, we can try using the low threshold voltage transistors for M5 ~ M8 transistors.

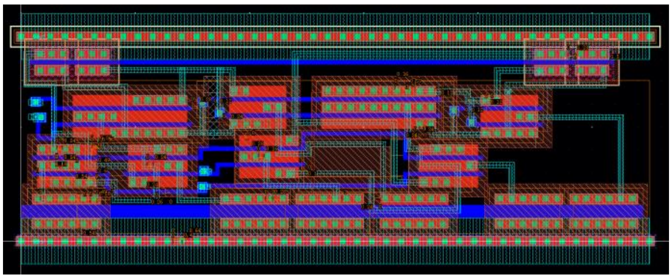
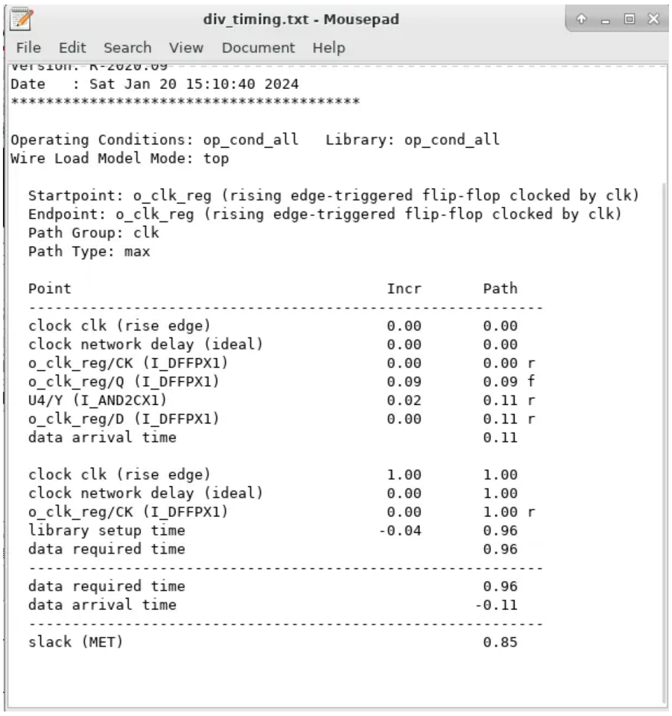


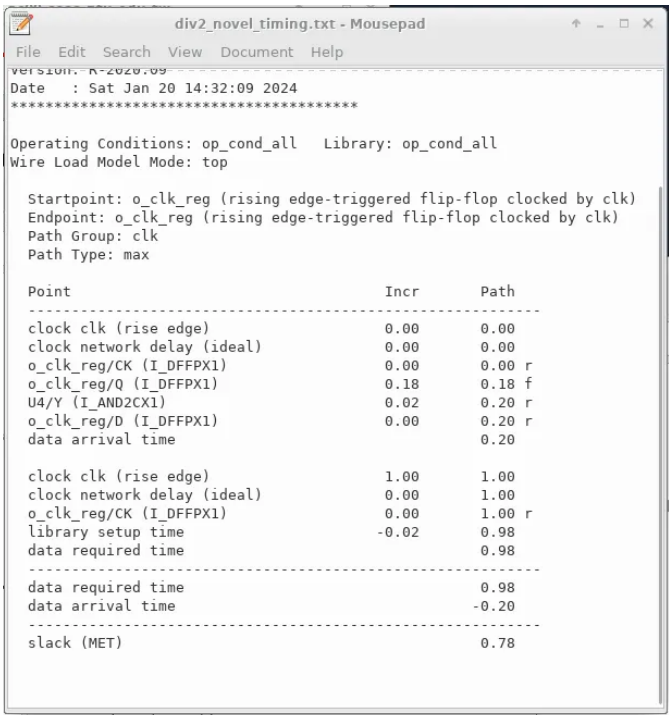
Fig.9. Novel Structure of High-Speed D Flip-Flop

5. RESULTS AND DISCUSS


div_timing.txt - Mousepad
File Edit Search View Document Help

```
Version: K-2020.09
Date   : Sat Jan 20 15:10:40 2024
****************************************

Operating Conditions: op_cond_all   Library: op_cond_all
Wire Load Model Mode: top

  Startpoint: o_clk_reg (rising edge-triggered flip-flop clocked by clk)
  Endpoint: o_clk_reg (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: max

  Point                                    Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  o_clk_reg/CK (I_DFFPX1)                  0.00       0.00 r
  o_clk_reg/Q (I_DFFPX1)                   0.09       0.09 f
  U4/Y (I_AND2CX1)                         0.02       0.11 r
  o_clk_reg/D (I_DFFPX1)                   0.00       0.11 r
  data arrival time                                   0.11

  clock clk (rise edge)                    1.00       1.00
  clock network delay (ideal)              0.00       1.00
  o_clk_reg/CK (I_DFFPX1)                  0.00       1.00 r
  library setup time                      -0.04       0.96
  data required time                                  0.96
  -----------------------------------------------------------
  data required time                                  0.96
  data arrival time                                  -0.11
  -----------------------------------------------------------
  slack (MET)                                         0.85
```

Fig.10. Original D Flip-Flop timing report


div2_novel_timing.txt - Mousepad
File Edit Search View Document Help

```
Version: K-2020.09
Date   : Sat Jan 20 14:32:09 2024
****************************************

Operating Conditions: op_cond_all   Library: op_cond_all
Wire Load Model Mode: top

  Startpoint: o_clk_reg (rising edge-triggered flip-flop clocked by clk)
  Endpoint: o_clk_reg (rising edge-triggered flip-flop clocked by clk)
  Path Group: clk
  Path Type: max

  Point                                    Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  o_clk_reg/CK (I_DFFPX1)                  0.00       0.00 r
  o_clk_reg/Q (I_DFFPX1)                   0.18       0.18 f
  U4/Y (I_AND2CX1)                         0.02       0.20 r
  o_clk_reg/D (I_DFFPX1)                   0.00       0.20 r
  data arrival time                                   0.20

  clock clk (rise edge)                    1.00       1.00
  clock network delay (ideal)              0.00       1.00
  o_clk_reg/CK (I_DFFPX1)                  0.00       1.00 r
  library setup time                      -0.02       0.98
  data required time                                  0.98
  -----------------------------------------------------------
  data required time                                  0.98
  data arrival time                                  -0.20
  -----------------------------------------------------------
  slack (MET)                                         0.78
```

Fig.11. Proposed High-Speed D Flip-Flop timing result

Although the proposed D flip flop is 2 times slower than the original one, we can confirm the topology works in the CML design flow. Further improvement can be implemented based on this topology.

The layout in Fig.13. is the decoder layout after the differential process is performed. It can be seen in the enlarged layout on the right that the original single-ended routing has been successfully converted to fully differential routing. Moreover, and the wires have been connected together without DRC/LVS error. Therefore, it can be concluded that our conversion is error free for small circuits albiet after numerous iterations redesigning the cells and modifying the single-ended design rules to trick the commercial CAD tools.

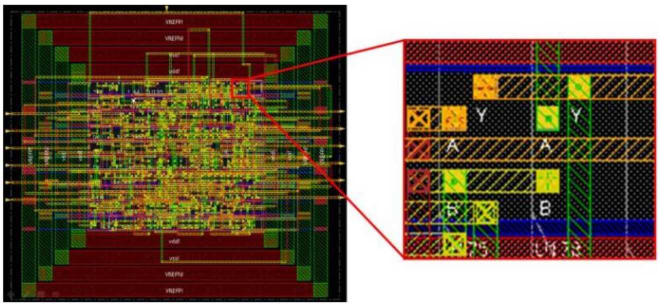


Fig.12. Single-ended result

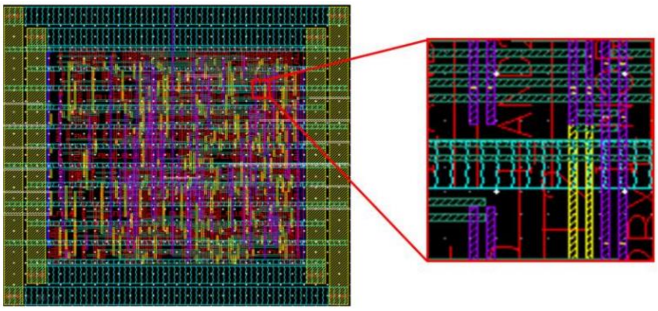


Fig.13. Differential result

4. CONCLUSION

This work improved the overall area and power consumption of the previous design. Additionally, a with an effort to design a high-speed DFF, we learned that a new D flip flop structure is tested to be functionable and requires further improvement. It is expected to improve the latency of the overall design with a tradeoff of area consumption. A functional MCML design flow is realized with the ability to accurately transform Verilog RTL into MCML layout and successfully tape out a chip using this flow. With such a design flow, designers that intend to design MCML circuits only need to focus writing the RTL code and develop high-performance MCML cells, which can significantly reduce the time and effort required for MCML circuit design. However, to reach the theoretical performance, we need to keep enhancing the MCML cell library.

5. REFERENCES

[1] Badel, Stéphane, Can Baltaci, Alessandro Cevrero, and Yusuf Leblebici. Design automation for differential MOS current-mode logic circuits. Springer International Publishing, 2019.

[2] Badel, Stéphane, Ilhan Hatirnaz, and Yusuf Leblebici. Semi-automated design of a mos current mode logic standard cell library from generic components. No. CONF. 2005.

[3] Badel, Stéphane, Ilhan Hatirnaz, Yusuf Leblebici, and Elizabeth J. Brauer. "Implementation of Structured ASIC Fabric Using Via-Programmable Differential MCML Cells." In 2006 IFIP International Conference on Very Large Scale Integration, pp. 234-238. IEEE, 2006.

[4] Hatirnaz, Ilhan, Stéphane Badel, and Yusuf Leblebici. "Towards a unified top-down design flow for fully differential logic blocks with improved speed and noise immunity." In Research in Microelectronics and Electronics, 2005 PhD, vol. 1, pp. 89-92. IEEE, 2005.

[5] Badel, Stéphane, Erdem Güleyüpoğlu, Özgür Inaç, Anna Pena Martinez, Paolo Vietti, Frank K. Gürkaynak, and Yusuf Leblebici. "A generic standard cell design methodology for differential circuit styles." In Proceedings of the conference on Design, automation and test in Europe, pp. 843-848. 2008.

[6] P. Heydari and R. Mohanavelu, "A 40-GHz Flip-Flop-Based Frequency Divider," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 53, no. 12, pp. 1358-1362, Dec. 2006.

[7] Xuelin Zhang, Yuan Wang, Song Jia, Ganggang Zhang and Xing Zhang, "A novel CML latch for ultra high speed applications," 2014 IEEE International Conference on Electron Devices and Solid-State Circuits, Chengdu, China, 2014, pp. 1-2.

[8] R. Mohanavelu and P. Heydari, "A novel ultra high-speed flip-flop-based frequency divider," 2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512), Vancouver, BC, Canada, 2004, pp. IV-169.

[9] T. Otsuji, M. Yoneyama, K. Murata and E. Sano, "A super-dynamic flip-flop circuit for broad-band applications up to 24 Gb/s utilizing production-level 0.2-/spl mu/m GaAs MESFETs," in IEEE Journal of Solid-State Circuits, vol. 32, no. 9, pp. 1357-1362, Sept. 1997.