Automatic logic synthesis for parallel alternating latches clocking schemes

D. Guerrero, M. Bellido, J. Juan, A. Millan, P. Ruiz, E. Ostua, J. Viejo
Dept. of Electronic Technology, University of Seville, Escuela Técnica Superior de Ingeniería Informática, Avda. de Reina Mercedes S/N, 41012 Sevilla, Spain

ABSTRACT

This paper proposes a VHDL coding technique that allows for the automatic synthesis of digital circuits using the so called Parallel Alternating Latches Clocking Schemes (PALACS). The proposed method greatly improves the applicability of PALACS and its benefits. This technique is verified through design examples in three different CMOS processes and using logic level simulation, with successful results in all the cases.

Keywords: Clock skew tolerance, high speed CMOS design, CAD circuit design

1. INTRODUCTION

VLSI digital systems have evolved to big and more complex systems being clocked at very high frequency. This evolution has reached a point that the overhead of the clock in the form of power consumption has become unacceptable. This is confirmed by what is observed in high-performance microprocessors[1]. So, reducing the power due to clock signal distribution is a mandatory issue in digital design. On the other hand, while the gate size and, as a consequence, the gate delay is getting smaller, the die size is rising. Since the delay in interconnection lines increases quadratically with the line length, it becomes longer than gate delay. Because of that the skew increases significantly. So, the simplest clocking scheme based on edge-triggered flip-flops should not be used for high-speed designs[2,3,4] as illustrated in Figure 1a: As we can see, if the clock skew is very long and the logic circuit is fast enough, the active edge of the clock can reach flip-flop 2 too late, i.e. near the instant when its input is going to change. Note that this problem can not be solved by enlarging the clock cycle[5]. To solve this problem, it has been suggested that the clock signal should reach first the registers at the end of the data path. Clock skew could cause malfunction anyway, as we can see in Figure 1b: If the clock skew is very long, flip-flop 2 could be triggered too early. This could be solved by enlarging the clock cycle, but re-routing the clock path is not a solution if feedback exists in the data path.

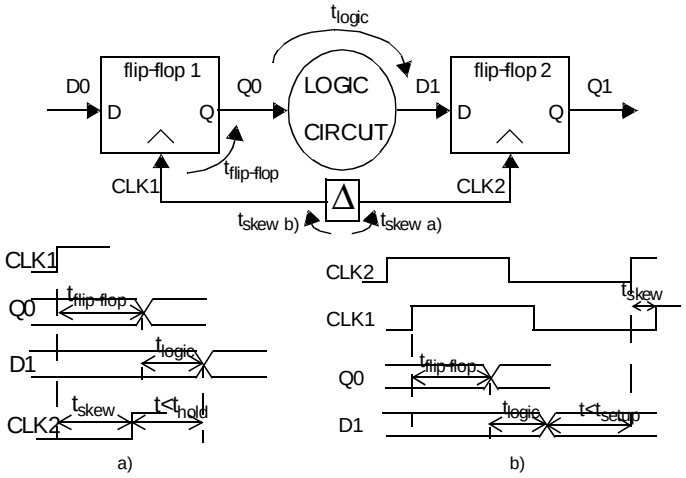


Fig. 1. Skew related problems in a single-phase system with flip-flops.

In order to prevent the clock skew from causing malfunction, a two-phase clocking scheme may be used. Two-phase clocking systems use two distinct clocks generated from the main clock at the last buffering stage. An example of two-phase clocking scheme is the two-phase Master-Slave clocking scheme (MSCS), which uses Master-Slave structures to implement the register block. A Master-Slave register working and its chronogram is shown in Figure 2, where it is assumed that the registers are transparent at the high level of the load signal.

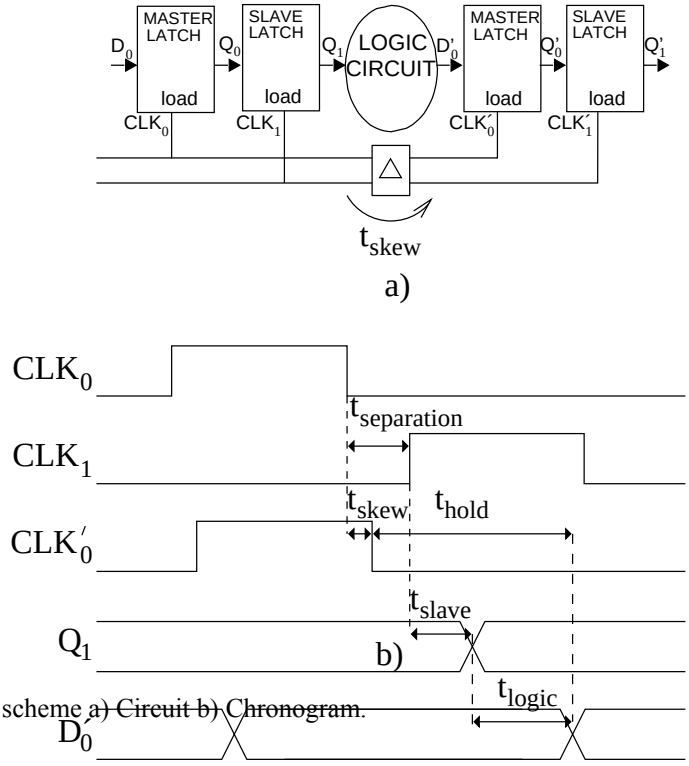


Fig. 2. Master-Slave clocking scheme a) Circuit b) Chronogram.

An alternative to MSCS are the Parallel Alternating Latches Clocking Schemes (PALACS)[6,7]. Like MSCS, PALACS provides skew tolerance by using multiple clock signals, but have remarkable advantages in power consumption and operation speed[6,7].

Besides, hardware description languages (HDL) are very convenient tools to design digital circuits, and logic synthesis software is commonly used to produce logic-level descriptions from high level HDL code. When coding a design in a HDL, the designer must follow a set of rules to ensure the description can be properly handled by the automatic synthesis tool. Thus, so-called canonical coding styles where combinational and sequential behaviour are clearly defined are preferred by logic synthesis tools, that usually deal with implementation details like the election of flip-flops and other logic blocks from the standard library provided by the technology files. In a full synchronous design, the usual coding techniques will typically produce single-edge-triggered flip-flops controlled by the same clock signal. In order to make use of PALACS in HDL descriptions, the sequential elements must be coded in a way that the logic synthesis tools can manage.

In this paper the authors describe VHDL coding techniques to automatically synthesise arbitrary circuits employing two-phase PALACS and four-phase PALACS. Targeting this objective, this paper is organised as follows: In the next section the PALACS clocking schemes will be summarised. In the following section a VHDL coding technique to describe circuits employing PALACS will be introduced. In the fourth section the correctness of these descriptions will be checked through automatic logic synthesis and logic-level simulation. Finally the conclusions summarised.

2. PARALLEL ALTERNATING LATCHES CLOCKING SCHEME

2.1 Two-phase PALACS

A remarkable alternative to the one-phase single-edge triggered flip-flop clocking scheme is the one-phase double-edge triggered flip-flop clocking scheme[8,9]. This scheme uses the flip-flop shown in Figure 3, that is triggered by both, falling and rising transitions of the clock signal. The power consumption of the clock distribution network in this scheme is smaller than using single-edge triggered flip-flops since there is an only clock transition per computation cycle.

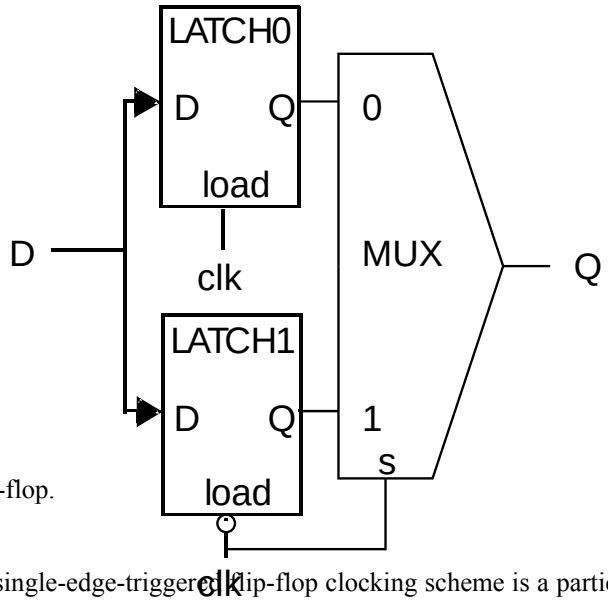


Fig. 3. Double-edge triggered flip-flop.

We could say that the one-phase single-edge-triggered flip-flop clocking scheme is a particular case of the MSCS where the slave clock signal is obtained by inverting the master clock signal, i.e. a particular case where the non-overlapping time between the clock signals is zero. The advantage of the general MSCS is that it provides tolerance to an arbitrary skew by enlarging the non-overlapping region.

In a similar way, the two-phase Parallel Alternating Latches Clocking Scheme (two-phase PALACS)[6] depicted in Figure 4 is a generalisation of the one-phase double-edge-triggered flip-flop clocking scheme. The memory element used in this scheme consists of two latches connected in parallel sharing the same input, and a switch at the output of each latch whose outputs are connected. The load terminals of both latches are controlled by separate phases, and the switches are also controlled by opposite phases. This scheme, unlike the Master-Slave scheme, allows reading and writing the register block simultaneously during the active level of each clock phase. When clock signal CLK0 is active, latch 0 loads the current input while latch 1 holds the previous input. The latch 1 data is read in the active phase of CLK0, since its switch is controlled by CLK0. When CLK0 becomes inactive, latch 0 stops being transparent. Then both phases remain inactive a time interval long enough to avoid clock-skew related problems. During this interval both switches are in high impedance (H.I.) state, but the previous data value remains loaded at the switches output due to parasitic capacitances. When CLK1 activates, the read-write mechanism works again, but both latches alternate their function, i.e. latch 1 loads a new value while latch 0 is read. We could say that this clocking scheme is the two-phase counterpart of the one-phase double-edge triggered flip-flop clocking scheme[6,7].

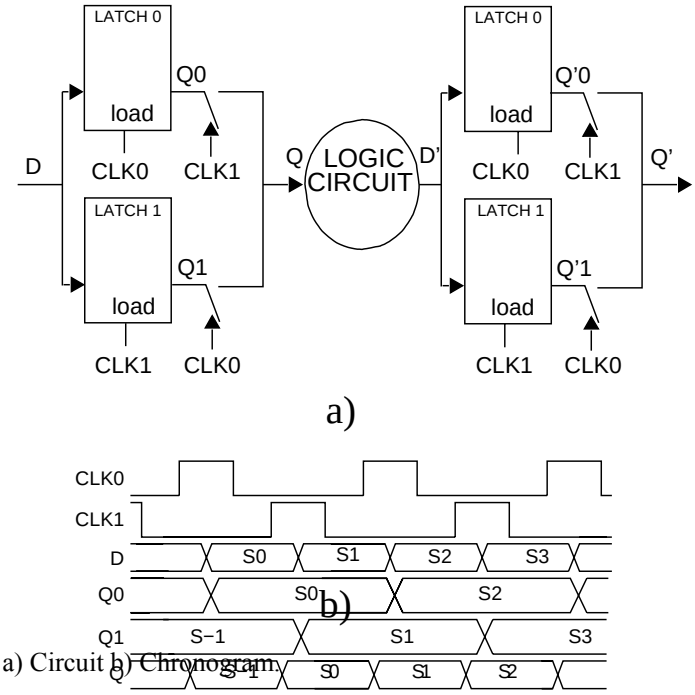
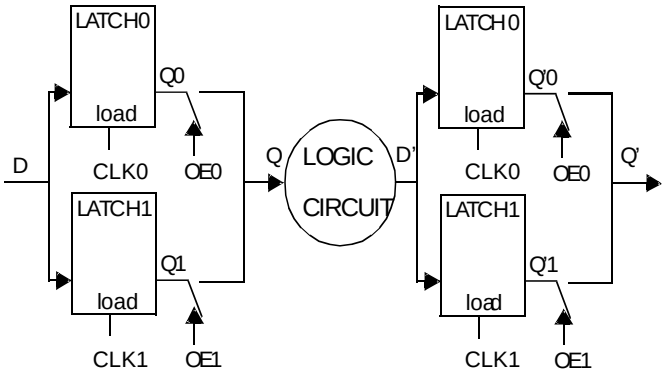


Fig. 4. Two-phase PALACS. a) Circuit b) Chronogram.

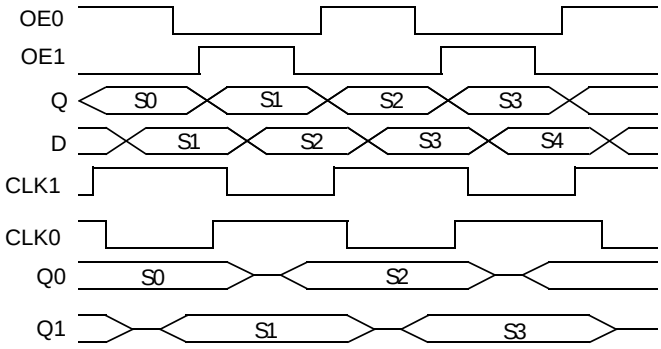
The most important advantage of PALACS versus MSCS is that the clock frequency is reduced by 50% for the same data rate. This has considerable benefits, mainly in the reduction of the power consumed by the clock distribution network. In PALACS, the number of clock transitions is two per computation cycle whereas in MSCS it is four. This means that their power dissipation can be reduced up to 50%. Another interesting advantage is that, for some implementations, the propagation delay of the PALACS structure is smaller than the propagation delay of the Master-Slave since with the MSCS the input signal has to propagate through two latches whereas in PALACS it has to propagate through one latch and a switch (whose delay is usually smaller than the delay of a latch). This produces an improvement in the operation speed of the system.

2.2 Four-phase PALACS

A drawback of the two phase PALACS is that the raising edges of the load control signals are hard edges[10]. This means that, regardless of the instant when a data item reaches a latch output, it will not keep propagating through the circuit until the load control signal of the opposite latch receive the next raising edge. In four-phase PALACS (Figure 5), the load control signals and the output enable control signals are not the same. So, a data item at the output of a latch can begin to propagate through the circuit even if that item has not been latched yet provided that the contamination delay of the logic circuit is long enough. So, the performance of the system can be improved by using time borrowing techniques[10] at the expense of using additional clock phases.



a)



b)

Fig. 5. Four-phase PALACS. a) Circuit b) Chronogram.

3. VHDL CODING TECHNIQUES FOR PALACS

As it was stated in the introduction, the possibility to describe digital circuits using PALACS using hardware description languages is essential to apply this clocking scheme in an extensive manner. The objective of this section is to present a VHDL coding technique for PALACS that is fully synthesizable by common logic synthesis tools, so that PALACS can be easily included in the standard digital design process. The process to code a design using PALACS can be divided in three steps:

- Description of a latch with tri-state output
- Description of the PALACS structure
- Coding the combinational part and instantiation of the PALACS structures

The first step, shown in Figure 6, is common to any design using PALACS and describes a single latch followed by a tri-state buffer, which is the basic building block for PALACS. Logic synthesis tools will typically render the structure in a single library latch with an output-enable controlling signal, or a latch plus a tri-state buffer, depending on what is available in the standard library.

```
entity latchOEclr is
  generic( n: integer:= 1);
  port (d,noe, ld, nclr: in std_logic;q: out std_logic);
end latchOEclr;
architecture behaviour of latchOEclr is
        signal qi: std_logic;
begin
  assign: process(ld, d, nclr)
  begin
    if ld='1' then
      qi<=d;
    end if;
    if nclr='0' then
      qi<='0';
    end if;
  end process;
  myoutput: process(qi,noe)
  begin
    q<='Z';
    if noe='0' then
```

Fig. 6. VHDL description of a tri-state output latch.

In The second step, which is also common to any design, two instances of the previous description are used to build up the PALACS structure using the structural VHDL description of Figure 7. The use of a structural description makes it possible to have better control over the automatic synthesis process and prevent the synthesis tools from changing the desired topology.

```
entity palacs4 is
  port (d, noe0, noe1, clk0, clk1, nclr : in std_logic;q : out std_logic);
end palacs4;
architecture mystruct of palacs4 is
  component latchOEclr
    port (d, noe, ld, nclr : in std_logic; q : out std_logic);
  end component;
begin
  latch0 : latchOEclr port map (d=>d,noe=>noe0,ld=>clk0,nclr=>nclr,q=>q);
```

Fig. 7. VHDL description of the PALACS structure.

In the third step, instances of the PALACS structure are added to the design in order to implement the sequential part. As an example, a rising four-bit counter is described in Figure 8. This description is similar to the canonical state machine coding style, where the process controlling the evolution to the next state has been substituted by the placement of a set of PALACS cells using a generate statement. Note that both, two-phase PALACS and four-phase PALACS can be implemented in this way, since two-phase PALACS is a particular case of four-phase PALACS where the output enable signals are the same that the load control signals.

```
entity cntMod16 is
  port (noe0,noe1,clk0,clk1,nclr: in std_logic;
    myoutput: out std_logic_vector (3 downto 0));
end cntMod16;
architecture mystruc of cntMod16 is
  signal std_cnt,nxt_std: unsigned (3 downto 0);
  component palacs4
    port (d,noe0,noe1,clk0,clk1,nclr: in std_logic;q: out std_logic);
  end component;
begin
  my_logic: process(std_cnt)
  begin
      myoutput<=std_logic_vector(std_cnt);
      nxt_std<=std_cnt+1;
  end process;
  generate_registers: for i in 3 downto 0 generate
```

Fig. 8. VHDL description of a four bit counter.

4. VERIFICATION OF THE VHDL DESCRIPTION STYLE FOR PALACS

In order to verify this VHDL coding technique, the description of the four-bit counter of Figure 8 has been used. Two-phase PALACS has been employed. To check the functionality, the circuit has been simulated at the logic level. The result is shown in Figure 9. As we can see, the counter works properly. Remarkably, the state signals are in high impedance when both output enable signals are disabled. This will not really occur due to parasitic capacitances[6,7].

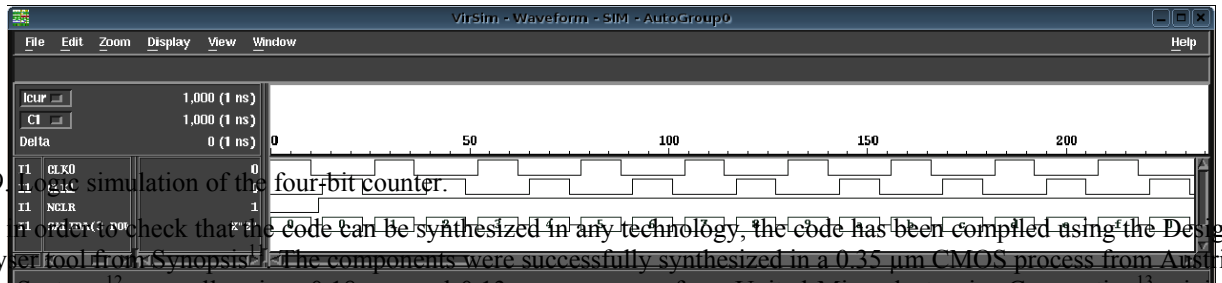
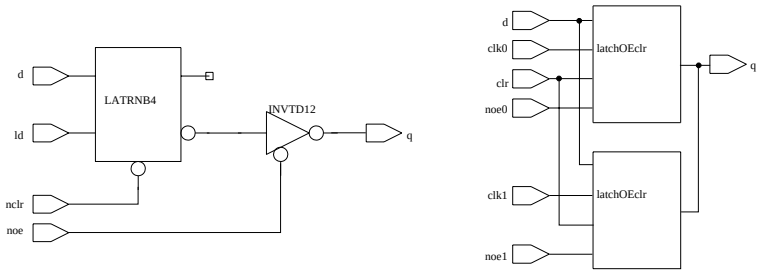


Fig. 9. Logic simulation of the four-bit counter.

Also, in order to check that the code can be synthesized in any technology, the code has been compiled using the Design Analyser tool from Synopsis[11]. The components were successfully synthesized in a 0.35 µm CMOS process from Austria Micro Systems[12], as well as in a 0.18 µm and 0.13 µm processes from United Microelectronics Corporation[13], giving similar results. As an example, Figure 10 shows the resulting implementation of the latch-buffer block (Figure 10a), and the PALACS structure (Figure 10b) for the 0.18 µm technology. It can be easily observed how the logic synthesis tool has selected appropriate components from the standard library provided by the foundry while keeping the desired functionality and topology for the PALACS structure. The netlist generated for the four-bit counter is depicted in Figure 11. Four PALACS structures has been placed as indicated in the VHDL description (Figure 8) and additional logic has been automatically synthesized to achieve the desired functionality.



a) b)

Fig. 10. Circuits synthesized by the tool a) Tri-state output latch b) PALACS structure.

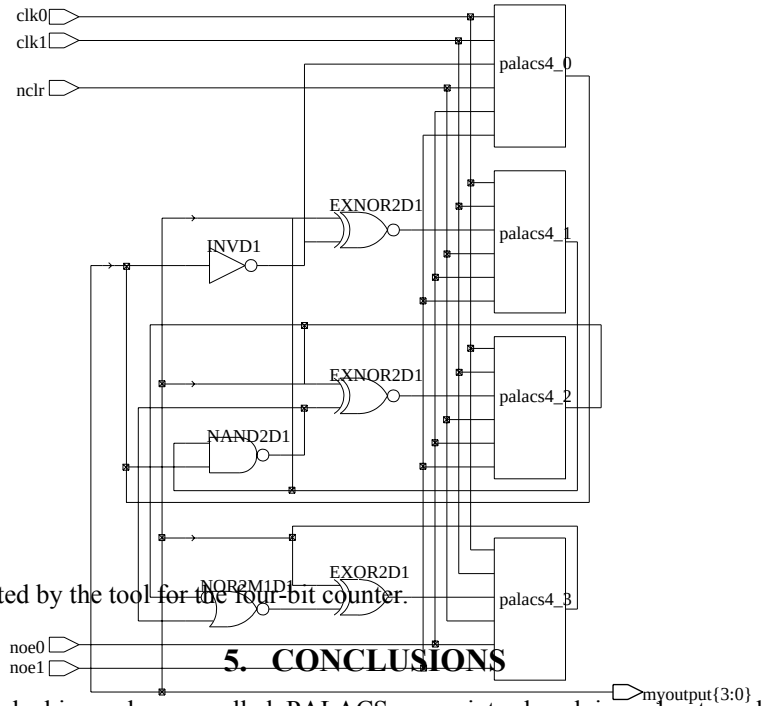


Fig. 11. The netlist generated by the tool for the four-bit counter.

5. CONCLUSIONS

In previous works, new clocking schemes called PALACS were introduced in order to solve clock skew related problems. In this paper, a VHDL coding technique is explored so that arbitrary sequential circuits can be automatically synthesized using PALACS. This technique is easily applied by re-defining the register block of the design. A sample VHDL design has been successfully implemented in three CMOS processes from two different foundries showing that the proposed VHDL descriptions are fully synthesizable and produce the right structures and behaviour, which has been checked through logic-level simulation. This result allow for a much wider and easier application of PALACS to general digital design.

ACKNOWLEDGEMENTS

This work has been partially supported by the Spanish Government's MEC META project TEC-2004-00840-MIC and the Andalusian Regional Government's CICE DHPMNDS projects EXC-TIC-1023 and EXC-TIC-635.

REFERENCES

1. V. Tiwari et al, “Reducing Power in High-Performance Microprocessors”, 35th Design Automation Conference, 1998, pp. 732-737
2. H. B. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*, Ed. Add-Wesley Publishing Company, 1990, ISBN 0-201-06008-6
3. K. Bernstein, *High Speed CMOS Design Styles*, Kluwer Academic Publishers, 1998, ISBN 0-7923-8220-X
4. S. H. Unger and CH. Tan, *Clocking Schemes for High-Speed Digital Systems*, IEEE transactions on computers , 1986, Vol. C-35. Nº10, pp. 880-895
5. M. Horowitz, “Clocking Strategies in High Performance Processors”, Symposium on VLSI Circuits Digest of technical Pagers, 1992, pp. 50-53
6. D. Guerrero, M. J. Bellido, J. J. Chico, P. Ruiz, A. Millan, “Two phase alternating latches clocking scheme for CMOS sequential circuits”, XVII Conference on Design of Circuits and Integrated Systems, November 2002, Santander, pp. 159-162
7. D. Guerrero, M. J. Bellido, J. J. Chico, P. Ruiz, A. Millan, E. Ostua, “Four phase alternating latches clocking scheme for CMOS sequential circuits”, XIX Conference on Design of Circuits and Integrated Systems, November 2004, Bordeaux
8. M. Afghahi and J. Yuan, “Double Edge-triggered D-flip-flops for High-speed CMOS circuits”, IEEE Journal of Solid-State Circuits, 1991, Vol. 26 Nº8, pp. 1168-1170
9. V. G. Oklobdzija, “Clocking and Clocked Storage Elements in Multi-GHz Environment”, 12th International Workshop PATMOS, 2002, pp. 128-145
10. D. Harris, *Skew-Tolerant Circuit Design*, Morgan Kaufmann Publishers, 2001, ISBN 1-55860-636-X, pp. 14-20
11. Synopsys Inc. http://www.synopsys.com/
12. Austria Micro Systems, http://www.austriamicrosystems.com
13. United Microelectronics Corporation, http://www.umc.com/