Automatic Synthesis of Non-Conventional Flip-Flops

Duarte Lopes de Oliveira **Henrique de Alencar Aguiar**
Departamento de Eletrônica Aplicada do Instituto Tecnológico de Aeronáutica
duarte@ita.br haaguiar@uol.com.br
Praça Marechal Eduardo Gomes, 50 - CEP 12228-900 - São José dos Campos - São Paulo - Brazil

ABSTRACT

The Flip-Flop (FF) memory element has a significant contribution in synchronous digital systems clock rate calculation. Due to the need of better the performance of a system digital, several techniques have been used for this purpose. An interesting technique is to use different types of (non-conventional) FFs resulting in a improvement in the performance of the digital system clock rate. In this article we propose a very practical method of automatic synthesis of the different types FFs. Our method steps eliminate of states minimization and state assignment. It synthesizes non-conventional FFs that will contribute in the increase of the system clock rate.

Keywords: flip-flops types, asynchronous logic, hazards, critical race and automatic synthesis.

1. INTRODUCTION

The increase synchronous digital systems applications and the need of better performance, many techniques had been proposed to Finite State Synchronous Machines (MEFS) and Datapath [1,2] synthesis. These techniques are related to low power [23,24], testability [21], high speed [5] and other logics [20]. In MEFS and Datapath synthesis the memory element Flip-Flop (FF) has a significant participation implementing many target architectures [1,2]. Targeting the performance increase of these architectures, new synthesis techniques, use each time more different types of FFs. Beyond conventional FFs (Single-Edge-Triggered (SET)$\rightarrow$D, T, JK and RS), there is a demand for FFs types like fuzzy [20], self-checking [15], Q [22], scan [16], self-timed [27], Bist [21], Double-Edge-Triggered [4,25], etc. For example, the usage of Double-Edge-Triggered FFs is an interesting alternative to power reducing and speed increasing [5]. Another way for target architecture performance increase, reducing the number of levels in critical path, so increasing clock rates, is to join the excitation logic with its respective FF [18][1]. In this article the different FFs types are called non-conventional FFs.

Many tools had been proposed to asynchronous circuit synthesis, like: Petrify [27], Assassin [28], Minimalist [29], 3D [6,7,8] and Miriã-GFM [14]. These tools can synthesize many FFs types, but the performance might be compromised because the synthesized FF must operate correctly (hazard free) and be optimized (area and setup, hold and propagation times). The tools Petrify and Assassin use the State Transition Graph (STG) specification, that is a Petri-Net interpreted [17,26] and the Minimalist tool uses Burst Mode (BM) specification, based in state diagrams [10]. The STG and BM specifications are not the most appropriate to specify the FF behavior, because cannot describe non-monotonic level sensitive signals (LSS – FFs inputs) [7, 26]. The 3D tool uses the Extended Burst-Mode (XBM) specification, that describes the FFs behavior, but the performance (area and speed) of FF-3D, based in basic gates, generated by the tool is inefficient [14]. The Miriã-GFM tool uses Multi Burst Graph [14] specification that is a XBM specification extension, generates many FFs types with good performance when compared to the FF-3D, but the LSS signals must satisfy the fundamental mode.

Because of asynchronous project problems and its performance, the synthesis of many FFs types is, in most cases, full custom, made manually and in a craft way [9,25]. In despite of these FFs normally having great performance, the manual and craft synthesis can be impracticable to most complex FFs types, what happens to the Merging FF [3,4]. In despite of full custom FFs have better performance, the FFs synthesized by basic gates and latches (standard-cell

[1] In this article we called the circuit to contain FF and the excitation logic of the type merging FF.

technology) are interesting too, because of cost and project time reducing. In [4] is presented a technique for conventional FFs synthesis from latches and basic gates, but the performance is impracticable (setup time too long).

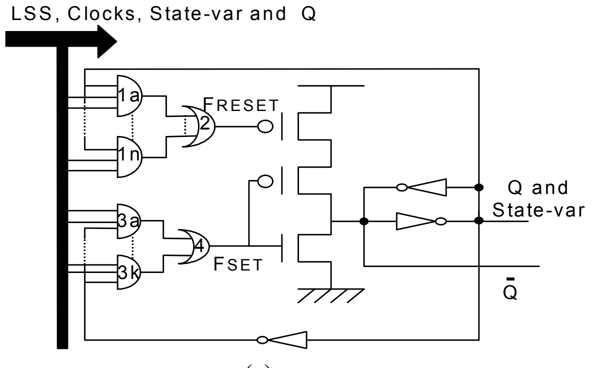
This articles proposes an automatic synthesis method to all FFs types, use the Asynchronous State Transition Graph (ASTG) specification that is a variant from GMR specification that permits efficiently describing any FF. The FFs are implemented in the target architecture based in basic gates and RS latches, called Feedback Set-Dominant (FSD-latch) that is a variant from architecture used by Petrify tool [27], or in full-custom target architecture known as gC (generalized C elements). Our method can synthesize FFs with multiple inputs and that realizes most several operations. These FFs can be activated by multiple clocks, have clock selection mechanisms, clocks working in both transitions and FFs that operate according to handshaking protocol. The FSD-latch architecture reduce the area and setup, latency and cycle times when compared to the FF-3D. To full-custom gC architecture the FFs obtained have great performance and are not synthesized manually and craft way.

Our method main characterisc is the elimination of asynchronous synthesis classical steps, like hazard free state minimization and critical run free states marking, what facilitates its understanding and usage. As illustration many FFs types are shown in the ASTG specification, and in our synthesis technique, in both target architecture, through a FF merging synthesis example.

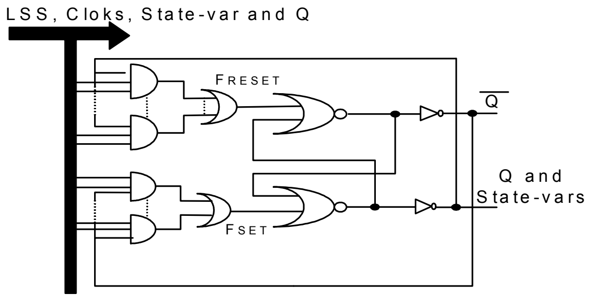
This paper is organized as follows: in the section 2, we present a temporal analysis of the proposed FSD-latch architecture; in the section 3, we present ASTG specification that is a variant of the MBG specification; in the section 4, we explain the method's synthesis procedures and present the extension for the Nowick's logic minimization theory. In the section 5, we present an example for demonstration; in the section 6, we compare the results from our method with the ones obtained with 3D, and finally, in the section 7, we present the conclusions and future work.

2. FSD-LATCH ARCHITECTURE

The Fedback-Set-Dominant latch architecture is a variant of the architecture presented in [28]. It is composed by FSET and FRESET functions (basic gates) and the RS latch (two configurations – to see figures 1a,b). The two feedbacks that arrive in FSET and FRESET functions, block the propagation of the glitches caused by the behavior non monotonic of the LSS signals. These feedbacks reduce the setup time of the FF. The figure 2 shows the gC full-custom target architecture that obtains a optimum propagation time with a medium cost in the area.



(a)



(b)

Figure 1 – Feedback set-dominant latch-FF architecture: a) hybrid (gates+transistors) latch; b) basic gates latch

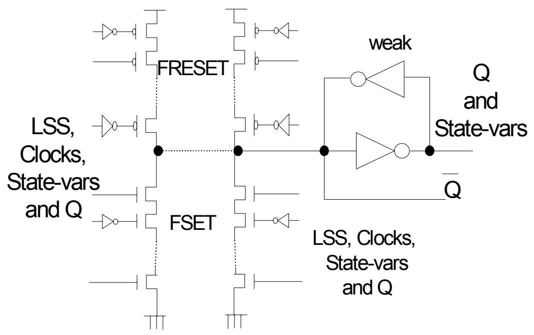


Figure 2 – gC architecture.

2.1 Timing analysis

Setup time is the minimal time interval between the LSS signals stabilization and the clock transition. *Hold time* is the minimal time interval between the clock signal transition and the allowed change of LSS signals in the next state transition. The hold and setup times are

implementation dependent (in our case, from the FSD-latch architecture).
From the FSD-latch architecture in figure 1a, we obtain the following time equations, where T is the Latch and logic gate delay [5,19]:

The minimum setup time is:

$$T_{SETUP} \geq T_{MAX\text{-}INV} \qquad (1)$$

The minimum hold time is:

$$T_{HOLD} \geq T_{MAX\text{-}AND\text{-}1a} + T_{MAX\text{-}OR\text{-}2} + T_{MAX\text{-}LATCH} - (T_{MIN\text{-}AND\text{-}1a} + T_{MIN\text{-}OR\text{-}2}) \qquad (2)$$

The maximum propagation time is:

$$T_{MAX\text{-}PROPAGATION} \geq T_{MAX\text{-}AND\text{-}1a} + T_{MAX\text{-}OR\text{-}2} + T_{MAX\text{-}LATCH} \qquad (3)$$

The minimum clock pulse width is:

$$T_{MIN\text{-}PULSE} \geq T_{HOLD} \qquad (4)$$

3. ASTG SPECIFICATION

The method begins from a proposed specification, called, as asynchronous state transition graph (ASTG). The ASTG specification is a variant of the burst-multi graph specification (BMG), that but allow OR and XOR causality between LSS and clock signals[2].
The ASTG specification is represented by a graph, in which vertices represent stable states while arcs represent state transitions. An initial state must exist. There are two types of signals: **1)** transition sensitive signals with monotonic behavior, which are the clock signals, state variables and output signals; clock signals can be mentioned in the state transition as don't-cares[3] (they can have the edge triggered of the clock signal or not); **2)** LSS signals with non-monotonic behavior that are input signals.
LSS signals that are not mentioned in a state transition have don't-care behavior. Clock signals, state variables and output signals that are not mentioned in a state transition keep their last value (they should not be activated). The ASTG allows the AND, OR and XOR causalities among signals LSS and clock signals.

The ASTG must satisfy the following rules:

[2]The MBG specification allows only OR causality among transition sensitive signals.
[3] Don't-care signal in ASTG is directed don't-care signal of the XBM specification [6,7,8].

1. The clock signals must obey a polarity sequence in the state transitions. For example, clk+ → clk- → clk+.... is acceptable, but clk+ → clk+ → clk-.... is not. To solve the polarity problem, we must introduce new state transitions. Clock don't-care signal doesn't necessarily obey the polarity sequence. For example, clk+→ clk* → clk* → clk+….
2. In all state transition, there should be a clock signal, sometimes denominated compulsory. A clock signal is compulsory if in the previous state transition the clock signal is not don't-care.
3. LSS signals mentioned in state transitions are labeled as a canonic (or minimum) function F or complementally F.
4. LSS signals mentioned in state transitions that emerge from the same state should be mutually exclusive F and F' complementally (restriction called as distinguishability condition in the XBM specification) [7,8].
5. The canonic function F with more than a term is described or with the OR operator either with the XOR operator

The figure 3 shows the general ASTG for conventional and non-conventional single-edge triggered flip-flop (SET-FF). The canonic function F describes the logic function that FF should execute and his F' complementally. The figure 4 shows the general ASTG for conventional and non-conventional double-edge triggered flip-flops (DET-FF).

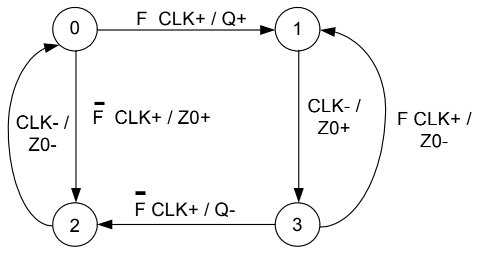


Figure 3 – General ASTG of the SET-FF

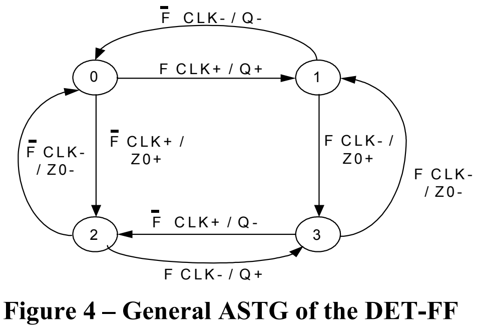


Figure 4 – General ASTG of the DET-FF

3.1 Examples: ASTG

As illustration, presented different types FFs that the Miriã-FF tool can synthesize. The F and F' functions are in the minimum form. The figures 5-11 show the ASTG of the FFs types. For example, the figure 7 shows ASTG of D-type SET-FF with two inputs (D_1 and D_2). This FF type reduces the excitation logic (sum-of-products) because eliminate the OR gate. The figure 8 shows the ASTG the D-type SET-FF with enable signal (H). This FF type was proposed for design for testability method that enhances the controllability of storage elements [21].

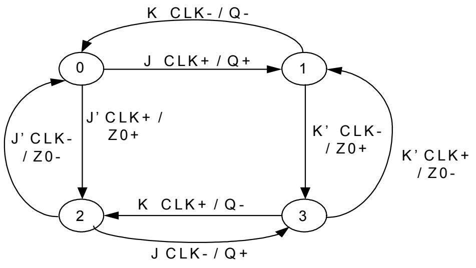


Figure 5 – ASTG of the JK-type DET-FF.

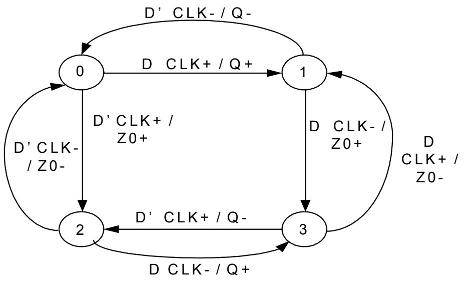


Figure 6 – ASTG of the D-type DET-FF.

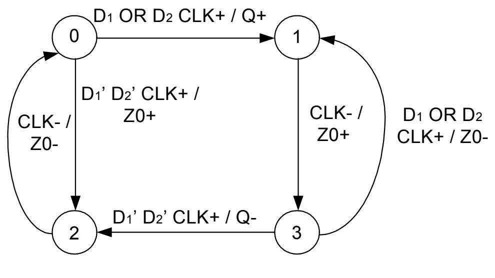


Figure 7 – ASTG of the D-type SET-FF with two inputs.

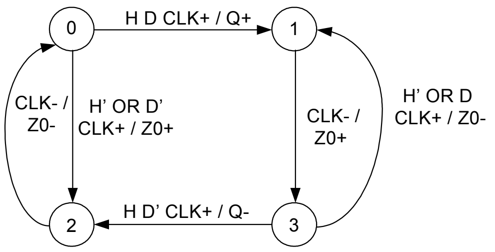


Figure 8– ASTG of the D-type SET-FF with clock enable.

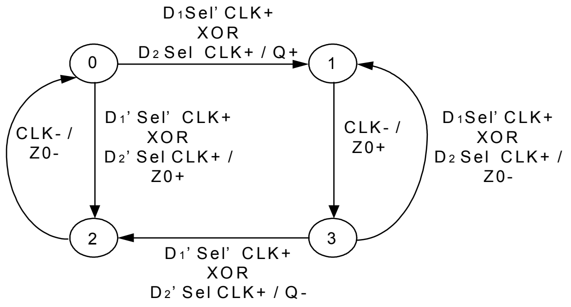

D1Sel' CLK+ XOR D2 Sel CLK+ / Q+
CLK- / Z0-
D1' Sel' CLK+ XOR D2' Sel CLK+ / Z0+
CLK- / Z0+
D1Sel' CLK+ XOR D2 Sel CLK+ / Z0-
D1' Sel' CLK+ XOR D2' Sel CLK+ / Q-
0
1
2
3

Figure 9 – ASTG of the D-type SET-FF with date select

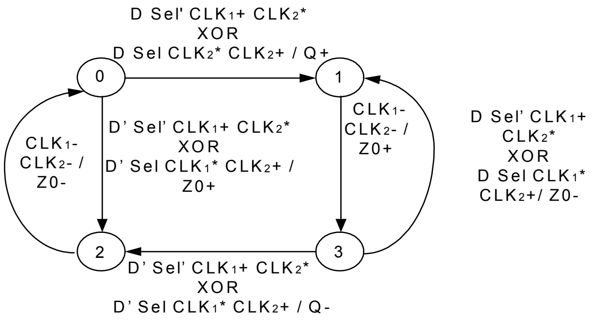

D Sel' CLK1+ CLK2* XOR D Sel CLK2* CLK2+ / Q+
CLK1- CLK2- / Z0-
D' Sel' CLK1+ CLK2* XOR D' Sel CLK1* CLK2+ / Z0+
CLK1- CLK2- / Z0+
D Sel' CLK1+ CLK2* XOR D Sel CLK1* CLK2+/ Z0-
D' Sel' CLK1+ CLK2* XOR D' Sel CLK1* CLK2+ / Q-
0
1
2
3

Figure 10 – ASTG of the D-type SET-FF with clock select

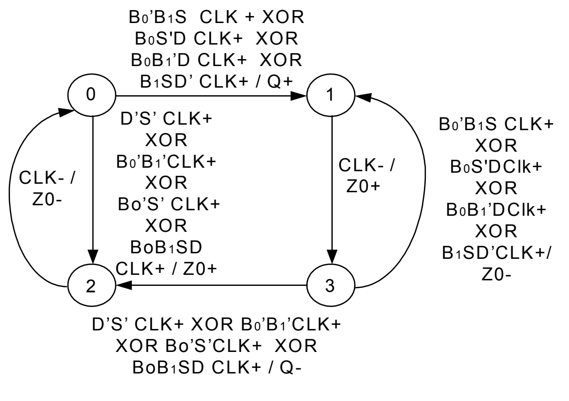

B0'B1S CLK + XOR B0S'D CLK+ XOR B0B1'D CLK+ XOR B1SD' CLK+ / Q+
CLK- / Z0-
D'S' CLK+ XOR B0'B1'CLK+ XOR Bo'S' CLK+ XOR BoB1SD CLK+ / Z0+
CLK- / Z0+
B0'B1S CLK+ XOR B0S'DClk+ XOR B0B1'DClk+ XOR B1SD'CLK+/ Z0-
D'S' CLK+ XOR B0'B1'CLK+ XOR Bo'S'CLK+ XOR BoB1SD CLK+ / Q-
0
1
2
3

Figure 11 – ASTG of the BIST-type SET-FF.

4. SYNTHESIS METHOD: PROCEDURES

The Miriã_FF tool performs the following tasks:

1. **To obtain ASTG:** Any type FF can be specified with four states in ASTG, therefore ASTG is already minimized and the assignment of the state variables free of critical race is trivial. In this step ASTG must capture the behavior of the type FF.
2. **To obtain STCs table:** transformation of the ASTG into a table of signals transition cubes [14].
3. **Two-level hazard-free logic minimization:** To each state variable and Q signal we must find the *Fse*t and *Freset* hazard-free sum-of-products functions. (to see section 4.1).
4. **Essential hazard analysis**: After logic minimization, delay elements may have to be inserted in order to avoid essential hazard [12]. An essential hazard is an undesirable race between an input (clock, LSS) and the signals Z0 or Q. It may be avoided by inserting a delay element in the feedback Z0$\rightarrow$Q or Q$\rightarrow$Z0.

4.1 Two-level Hazard-free logical minimization

A modified version of the literal exact ATACS algorithm [13], that we called ATACS-FF and were used for the logic minimization task. The two modifications are:

1. ATACS-FF explores a table of STCs instead of a compacted state graph.
2. In order to guarantee that the resulting circuits will be logic hazard free: ATACS-FF should satisfy 2 requirements: Lemma 4.1 and Theorem 4.1. They replace the requirements proposed by Nowick [10,11] and Yun [7,8].
   - 2.1. All trigger cubes free of violations (prime implicants that cover a required cube) should satisfy Lemma 4.1 (when building and solving the binate table).
   - 2.2. The cover of function $F_{SET\text{-}(Z0,Q)}$ $(F_{RESET\text{-}(Z0,Q)})$ (solution of the unate table) should satisfy the theorem 4.1

Lemma 4.1 (without proof) Let $f_{(Z0,Q)}$ represent Boolean function of the state variable *Z0* or output Q. The $F_{SET\text{-}(Z0,Q)}$ $(F_{RESET\text{-}(Z0,Q)})$ represents the correspondent sum-of-products implementations of the *Set (Reset)* signals of the feedback set-dominant latch architecture. Let $STC_1[I_T, O_T]$ represent a t_S transition, $STC_2[I_{T1}, I_{T2}, O_{T1}, O_{T2}]$ a t_{OR} transition and STC_3 $[I_{T1}, I_{T2} \ldots I_{TN}, O_{T1}, O_{T2}, \ldots \ldots O_{TN}]$ a t_{XOR} transition.

1. If $f_{(Z0,Q)}$ presents a transition $0 \rightarrow 0$ in or STC_1 either STC_2 either STC_3, then $F_{SET\text{-}(Z0,Q)}$ is logic hazard-free (non covering exists).
2. If $f_{(Z0,Q)}$ presents a transition $1 \rightarrow 1$ or $1 \rightarrow 0$ in or STC_1 either STC_2 either STC_3, then $F_{SET\text{-}(Z0,Q)}$ is logic hazard-free because non cover is needed ($f=1$ $\rightarrow$don't-care states).
3. If $f_{(Z0,Q)}$ presents a transition $0 \rightarrow 1$ in or STC_1 either STC_2 either STC_3, then $F_{SET\text{-}(Z0,Q)}$ is logic hazard-free if and only if there is a product $p_i \in F_{SET\text{-}(Z0,Q)}$ that completely covers O_{Ti} and if there is a product $p_j \in F_{SET\text{-}(Z0,Q)}$ such that $p_i \cap p_j \neq \varnothing$ then p_j cover the final state.

There is a similar lemma for the F_{RESET} function.

Theorem 4.1 (proof [10]): A cover of function $F_{SET\text{-}(Z0,Q)}$ $(F_{RESET\text{-}(Z0,Q)})$ is logic hazard-free if and only if:

1. No product of $F_{SET\text{-}(Z0,Q)}$ $(F_{RESET\text{-}(Z0,Q)})$ crosses the ON-set (OFF-set) of $f_{(Z0,Q)}$.
2. Each required cube of $f_{(Z0,Q)}$ is completely contained in a unique product of $F_{SET\text{-}(Z0,Q)}$ $(F_{RESET\text{-}(Z0,Q)})$.
3. All products $p_i \in F_{SET\text{-}(Z0,Q)}$ $(F_{RESET\text{-}(Z0,Q)})$ satisfy Lemma 4.1.

5. Example

The figure 12 shows a part of a synchronous state machine. This example will be used to illustrate our technique. It will synthesize the SET-FF of the type merge (SET-FF-D + excitation logic). The figure 13 shows new merge FF with 4 inputs that correspond to the inputs (ACK,Q2,Q1,Q0) of the excitation logic. The first step is to extract the canonic function F and his complement, obtained of the excitation logic of the figure 12.

$$F\ (D0,D1,D2,Dack,Z0)=\sum\ (1,3,4,5,7,10,11,12,15)$$

e

$$F'\ (D0,D1,D2,Dack,Z0)=\sum (0,2,6,8,9,12,13)$$

The second step is to obtain ASTG for the SET-FF merge. The tool Miriã-FF accepts ASTG described in a textual language [14].

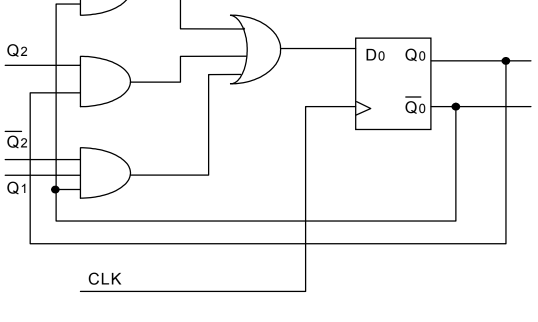


Figure 12 – Part of a synchronous state machine.

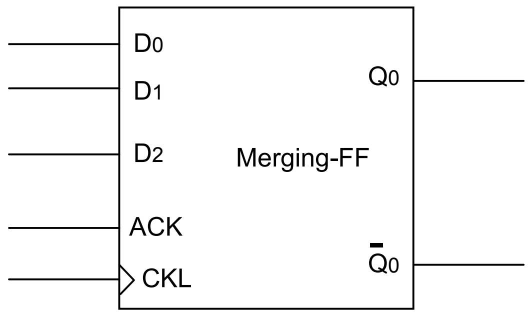


Figure 13 – Merging SET-FF.

After logic minimization, the Miriã-FF tool produced the SET and RESET functions for the output Q and state variable Z0

Q_{SET}= CLK D0 D2'Z0' + CLK D0'DACK Z0' + CLK D0'D1 D2'Z0'
Q_{RESET}= CLK D0 D2'ZO + CLK DO'D2 DACK'Z0 + CLK DO'D1'DACK'Z0

$Z0_{SET}$= CLK'Q + CLK DO D2'Q' + CLK D0'D2 DACK'Q' + CLK D0'D1'DACK'Q'
$Z0_{RESET}$=CLK'Q' + CLK D0 D2 Q CLK D0'DACK Q + CLK D0'D1 D2'Q

Figures 13 and 14 shows the netlist resulting from a trivial technology-mapping for the respectively feedback set dominant latch and gC architectures

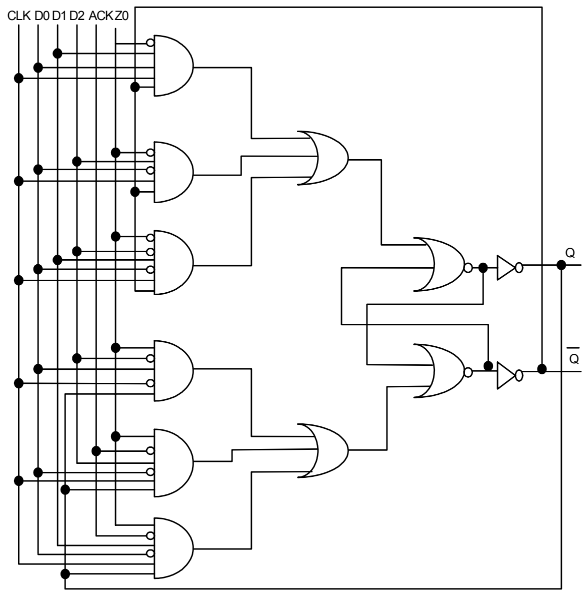


Figure 14 – FSD-latch circuit of the Q signal.

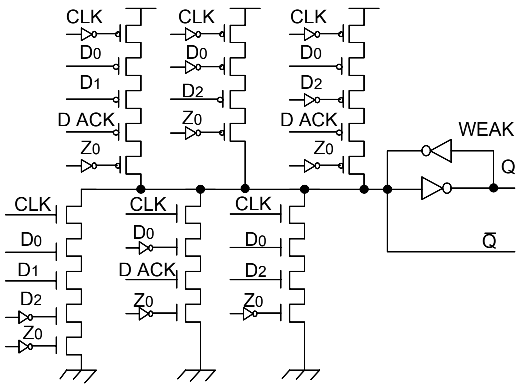


Figure 15 – gC circuit of the Q signal.

6. RESULTS AND DISCUSSION

In this section we compare our results using CMOS technology standard-cell library of the IMEC-96 of 0.07μm. The comparison is made in terms of number of transistors and (setup + propagation times). The tables 1,2 and 3 show the synthesis results of the Miriã-FF-FSD-latch for different FF types. The table 1 shows 5 (home made) benchmark examples of the SET-FF types. The results of our tool shown in table 1 obtain a average reduction of delay (ts + tp) of 20% with penalty of 600% in the number of transistors. The table 2 shows 5 (home made) benchmarks examples of the DET-FF types[4]. The results are of the 11% in the average reduction of delay with penalty of 850% in the number of transistors. The table 3 shows 6 examples of known FF types[5]. The results are of the 60% in the average reduction of delay with penalty of 58% in the number of transistors. The table 4 shows the 16 examples implemented in the gC full custom architecture with average penalty of 29% in the number of transistors, but these FFs operates in the average delay of 1ns (ts+tp).

	In	Miriã-FF-FSD-Latch		Standard Logic	
		Nt	Tp+Ts (ns)	Nt	Tp+Ts (ns)
OPL-SET-FF	3	432	3.74	48	4.29
BIST-SET-FF [21]	4	484	3.74	62	4.66
CSTP-SET-FF [21]	4	362	3.36	54	4.29
MUX-SET-FF (Shift-Register)	3	294	3.26	36	4.03
Merging-SET-FF	4	282	3.07	46	4.29

Symbols: In (input signals); Nt (number of transistors); Ts (max setup time); Tp (max propagation time)

Table 1 - Results SET-FF: Miriã-FF-FSD-Latch and Standard-cell logic.

4 The D-DET FF (using standard-cell library) was synthesized with two D-SET-FFs + mux 2x1 [24].
5 The used standard-cell library only the D-SET is available.

	In	Miriã-FF-FSD-Latch		Standard Logic	
		Nt	Tp+Ts (ns)	Nt	Tp+Ts (ns)
OPL-DET-FF	3	826	4.67	86	4.85
BIST-DET-FF [21]	4	954	4.81	100	5.22
CSTP-DET-FF [21]	4	712	4.28	92	4.85
MUX-DET-FF (Shift-Register)	3	570	4.04	73	4.59
Merging-DET-FF	4	504	3.88	84	4.85

Table 2 - Results DET-FF: Miriã-FF and Standard-cell logic.

	In	Miriã-FF-FSD-Latch		Standard Logic	
		Nt	Tp+Ts (ns)	Nt	Tp+Ts (ns)
D-DET-FF	1	70	1.83	26	3.47
D-DET-FF	1	96	1.93	66	4.03
JK-SET-FF	2	82	1.83	38	4.03
JK-DET-FF	2	100	1.93	90	4.59
T-SET-FF	1	80	1.83	38	4.03
T-DET-FF	1	96	1.93	90	4,59

Table 3 - Results Known FF: Miriã-FF and Standard-cell logic.

	Miriã-FF-gC (Nt)
D-SET-FF	32
D-DET-FF	40
JK-SET-FF	34
JK-DET-FF	42
T-SET-FF	32
T-DET-FF	40
OPL-SET-FF	88
OPL-DET-FF	146
BIST-SET-FF [21]	112
BIST-DET-FF [21]	174
CSTP-SET-FF[21]	78
CSTP-DET-FF[21]	124
MUX-SET-FF	64
MUX-DET-FF	100
MERGING-SET-FF	78
MERGING-DET-FF	136

Table 4 – Results FF types: gC architecture

7. CONCLUSIONS

For digital systems that need high-speed and uses standard-cell technology or full custom either programmable devices (by example PLA) the conventional FFs are an obstacles. The non-conventional FFs synthesis is an interesting alternative for to increase speed of the digital systems. This article proposes the specification denominated asynchronous state transition graph (is a variant of mult-burst graph) and FSD-latch architecture as a standard architecture to flip-flops. We have demonstrated that, when using the FSD-latch architecture, we can synthesize non-conventional FFs with a superior performance in speed, when compared with the part of the circuits (FF + excitation logic) generated by the standard-cell library. As next steps in future works, we will to research the possibility of reduction of transistors.

REFERENCES

[1] E. J. MacCluskey, Logic Design Principles With Emphasis on Testable Semicustom Circuits, Prentice-Hall, 1986

[2] R. H. Katz, Contemporary Logic Design, The Benjamin/ Cummings Publishing Company, Inc., 1994

[3] S. H. Unger, Asynchronous Sequential Switching circuits, Wiley-Interscience, John Wiley & Sons, Inc., New York, 1969.

[4] S. H. Unger, "Double-Edge-Triggered Flip-Flops", IEEE Trans. on Computers, vol. C-30, No. 6, June 1981, pp.447-451.

[5] S. H. Unger, "Clocking Schemes for High-Speed Digital Systems", IEEE Trans. on Computers, vol. C-35, No. 10, October, 1986, pp.880-895.

[6] K. Y. Yun, *Synthesis of Asynchronous Controllers for Heterogeneous Systems*, PhD thesis, Stanford University, 1993.

[7] K. Y. Yun and D. L. Dill, "Automatic Synthesis of Extended *Burst-Mode* Circuits: Part I (Specification and Hazard-Free Implementation)," IEEE Trans. on CAD of Integrated Circuit and Systems, Vol. 18:2, February 1999, pp. 101-117.

[8] K. Y. Yun and D. L. Dill, "Automatic Synthesis of Extended *Burst-Mode* Circuits: Part II (Automatic Synthesis)," IEEE Trans. on CAD of Integrated Circuit and Systems, Vol. 18:2, February 1999, pp. 118-132.

[9] K. Y. Yun, et al., "High-performance two-phase micropipeline building blocks: double edge-triggered latches and burst--mode select and toggle circuits," IEE Proc. Circuits Devices Systems, vol.143, no5, October, 1996, pp.282-288.

[10] S. M. Nowick, "*Automatic Synthesis of Burst-Mode Asynchronous Controllers*,"Ph.D. thesis, Stanford University, 1993.

[11] S. M. Nowick and D. l. Dill, "Exact Two-Level Minimization of Hazard-Free Logic with Multiple-Input Changes," IEEE Trans. on CAD of Integrated Circuits and systems, vol. 14, No. 8, August, 1995, pp.986-997.

[12] D. B. Armstrong, A. D. Fredman and P. R. Menon,"Realization of Asynchronous Sequential Circuits Without Insert Delay Elements," IEEE Trans. on Computers, vol. C-17, No.2, February 1968, pp.129-134.

[13] H. M. Jacobson and C. J. Myers, "Efficient algorithms for exact two-level hazard-free logic minimization," IEEE on Trans. CAD of integrated Circuits and Systems, vol.21, Nro.11, November 2002, pp 1269-1283.

[14] D. L. Oliveira, "*Miriã: uma ferramenta para síntese de controladores assincronos multi-rajada*," Tese de Doutorado, EPUSP, 2004.

[15] S. M. Kia, "Designs for self-checking flip-flops," IEE Proc. Comput. Digit. Tech. Vol. 145, Nro.2, March, 1998, pp.81-88.

[16] I. Ghosh and S. Bhawmik, "A Practical Method for Selecting Partial Scan Flip-flops for Large Circuits,"10th Int. Conf. on VLSI Design, January, 1997, pp.284-288.

[17] D. F. Cox, "D, T and JK Constraints in Asynchronous Synthesis," 8th NASA Symposium on VLSI Design, October, 1999.

[18] S. Sakaidani, et. Al. "Flexible Processor Based on Full-Adder / D-Flip-Flop Merged Module," xxx 2001, pp.35-36

[19] I. Vasiltsov, et. Al., "Estimation of the Reliability, Speed and Cost Parameters of the Different Type of Flip-Flops," I. Workshop on Intelligent Data Acquisitions and Advanced Computing Systems Technology and Applications, 2001, pp103-106.

[20] K. Hirota and K. Ozawa, "Concept of a fuzzy flip-flop," IEE Trans. systems Man and Cybernetics, Vol. 19, Nro.5, 1989, pp.980-997.

[21] C. E. Stroud, "Automated Bist for Sequential Logic Synthesis,"IEEE Design & Test of Computers, December 1998, pp.22-32.

[22] F. U. Rosenberger, et. Al., "Q-Modules: Internally Clocked Delay-Insensitive Modules," IEEE Trans. on Computers, Vol.37, Nro.9, September 1988, pp.1005-1018.

[23] T. Lang, et. Al., "Individual Flip-Flops with Gated Clocks for Low Power Datapaths," IEEE Trans. on Circuits and Systems-II: Analog Digital Signal, processing, Vol.44, Nro.6, June, 1997, pp.507-516.

[24] A. G. M. Strollo, et. Al., "Power dissipation in One-Latch and Two-Latch Double Edge Triggered Flip-Flops," Proc. 14th IEEE Int. Electronics, Circuits and Systems, vol 3, 1999, pp.1419-1422.

[25] M. Pedram, et al., "A New Design of Double Edge Triggered Flip-Flop," Proc. of the ASP-DAC South Pacific, 1998, pp.417-421.

[26] C. Piquet and J. Zahnd, "STG-Based Synthesis of Speed-independent CMOS Cells," Workshop on Exploitation of STG-based Design Technology, St. Petersburg, July, 1998.

[27] J. Cortadella, et al., "Petrify: a tool for manipulating concurrent specifications and synthesis of asynchronous controllers," IEICE Trans. on Inf. And Systems, vol.E80-D, no3, March, 1997, pp.315-323.

[28] C. Ykman-Couvreur, et al., "Assassin: A Synthesis for Asynchronous Control Circuits," Tech. Rep. IMEC, User and Tutorial manual, September, 1994.

[29] R. M. Fuhrer, S. M. Nowick et al., "MINIMALIST: An environment for the synthesis and verification of burst-mode asynchronous machines," Proc. IEEE/ACM Int. Workshop logic Synthesis, 1998.