A Synchronous Wrapper for Asynchronous Pipeline Modules in a Synchronous Design in FPGAs

Duarte L. Oliveira[1], Kledermon Garcia[1,2], Lester A. Faria[1], Higor A. Delsoto[1], Leonardo Romano[3]
[1]Electronic Engineer Division – Technological Institute of Aeronautics – ITA – IEEA
[2]Aeronautical Systems Division – Institute of Aeronautics and Space – IAE-CTA
[3]Electrical Engineer Department - University Center of FEI

***Abstract*—Taking advantage of both synchronous and asynchronous paradigms, a new style of design, called Globally Synchronous Locally Asynchronous (GSLA), has obtained very interesting results. In this paper, we propose a synchronous wrapper that allows the communication of asynchronous modules with a synchronous environment. The proposed synchronous wrapper comprises a locally asynchronous pipeline module. This asynchronous pipeline style shows to be interesting for FPGA platforms due to the simplicity of its controller. Through a case study, a 5th order FIR filter, it is shown that the proposed wrapper presents a reduction of 4% in the power consumption when compared with a synchronous pipeline design. The synchronous wrapper allows the asynchronous pipeline modules to interact with other synchronous modules up to a frequency of 500MHZ. For the case study, the synchronous wrapper provided an increase of 200% in global clock rate.**

***Keywords*—XBM specification, AFSM, logic asynchronous, GSLA**

I. INTRODUCTION

Highly complex electronic systems are more and more common in the aerospace sector. This sector is in a wide expansion in different areas, such as: satellites, rockets, missiles, aircraft, etc., what demands narrow requirements that hinder the project. These electronic systems must necessarily be based on the concept of "System-on-Chip - SoC".

The main reason is to satisfy a ruthless demand for high performance, reusability and low power requirements [1,2]. SOC circuits are characterized by integrating different kinds of functional modules of a computer system in a single chip, which may be “Intellectual Property (IP)” from different companies. These IP modules are pre-designed, checked and tested to achieve high performances. They allow reducing cost and, especially, time.

If SoC circuits are implemented in VLSI (Very Large Scale Integration) with a global clock signal, not only the performance and power (clock skew and distribution network) are affected, but also the timing analysis becomes more complex [3,4]. On the other hand, SoC circuits can also be implemented in FPGAs (Field Programmable Gate Array), although worsening the problem of clock skew. The main reason is the distance between macro-cells, leading to significant delays [5].

Duarte L. Oliveira, duarte@ita.br, Tel. +55-12-3947-6813; Kledermon Garcia, kledermonkg@iae.cta.br; Lester A. Faria, lester@ita.br; Higor A. Delsoto, higordel@hotmail.com; Leonardo Romano, leoroma@uol.com.br.

SoC circuits, in the aerospace environment, are easily found as embedded projects. Embedded digital systems (EDS) are computational systems, which may be composed by softwares and hardwares (e.g., ASIC - Application Specific Integrated Circuits), presenting specific design requirements related to the environment where they will work on.

Comparing FPGA and VLSI designs, we can say that the FPGA design leads to a much lower performance and to much higher power consumption, when compared to VLSI style. Due to the structure based on macrocells of FPGA devices, it is common to get a very low global clock rate for SoC systems, what can derail the application. There are several reasons for that, such as the significant delay of the macrocells, the communication delay between macrocells, and clock skew problem that, in FPGAs devices, is even more serious. One reason that strongly contributes for the global clock rate of a SoC system is that the overall rate is defined by the lower clock rate module. Several alternative solutions were proposed for the design in an FPGA or in a VLSI platform. Anyone of them can be chosen, aiming to eliminate the problems associated with the clock signal [6].

An alternative solution, highly widespread for increasing the performance of global clock SoC systems, is the GALS methodology (Globally Asynchronous Locally Synchronous). The term GALS was first used by Chapiro, in his PhD thesis [7]. A GALS architecture consists of many synchronous functional modules that communicate with each other in an asynchronous way, thus allowing multiple clocks, i.e., each functional module has its own local clock. GALS systems is an interesting solution, but a main drawback is the cost of communication between locally synchronous modules, and the cost of the construction of asynchronous wrappers, which is associated with each synchronous module.

Another alternative is availing the qualities of the asynchronous paradigm by introducing asynchronous modules in a synchronous platform/environment based on a global, or multiple clocks. Therefore, it is possible to benefit from the advantages of asynchronous paradigm, such as: *a)* asynchronous module does not interfere/reduce the system frequency, which is defined by the slower synchronous module; *b)* reduces the power of the clock signal, due to the reduction of the clock's activity in registers; and *c)* reduce the clock skew problem, because it reduces the need for buffers.

Therefore, taking advantage of synchronous and asynchronous paradigms, Sjogren, et al. [8] proposed a new design style called Globally Synchronous Locally Asynchronous (GSLA), which allows the interaction between

synchronous and asynchronous modules. In addition, it reduces power consumption, improves the performance of the system, and reduces electromagnetic interference, buffers and noise, as well as reduces the clock skew.

Different levels of GSLA have been proposed in literature. In [9], an asynchronous instruction decoder was inserted in the synchronous processor, from Intel, achieving a high performance and a high power reduction. In [10], a high performance synchronous filter uses, internally, an asynchronous pipeline [11]. And finally, in [12-18], different "synchronous-asynchronous-synchronous" interfaces have been proposed, but all of them leading to full-custom projects. Figure 1 shows a multi-point system (GSLA type) composed by synchronous and asynchronous modules. It is synchronized by a global clock and can have multiple clocks.

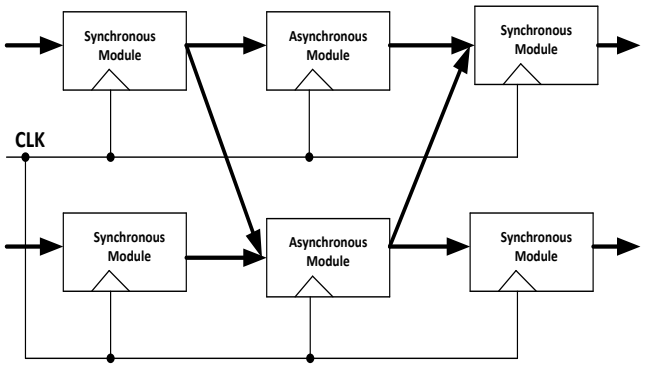

Fig. 1. Modules asynchronous/synchronous with clock global.

The interaction between asynchronous and synchronous modules in GSLA style needs a synchronization of the asynchronous module, thus needing a synchronous wrapper, as shown in Fig. 2. Considering the asynchronous systems design style, it is easy to implement the asynchronous pipeline style in commercial FPGAs, due to the simplicity of its controller [4, 5]. This is important because of the difficulty to achieve hazard-free asynchronous controllers, especially when they are implemented in these platforms. Therefore, this paper focuses on locally asynchronous modules, which are implemented in the pipeline style (see Fig. 3).

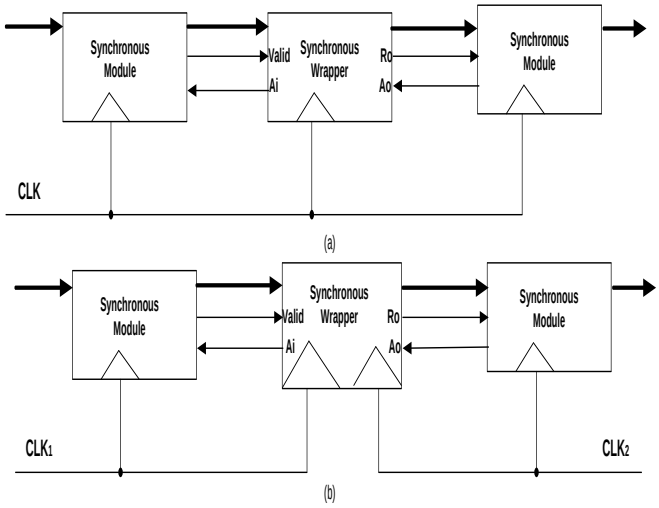

Fig. 2. Schemes: a,b) synchronous to asynchronous to synchronous.

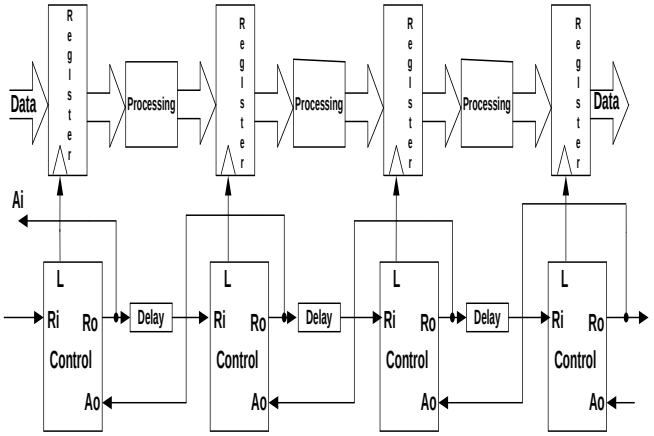

Fig. 3. Linear asynchronous pipeline architecture of [19].

This paper proposes a novel synchronous wrapper for GSLA systems, as shown in Fig. 4. The synchronous wrapper contains a locally asynchronous pipeline module and two communication ports that allow a high-performance interaction with other synchronous modules. The asynchronous pipeline module is designed in the same architecture proposed in [19]. Through a case study, a 5th order FIR filter, it is shown that the proposed wrapper provides a 200% increase in the overall clock rate.

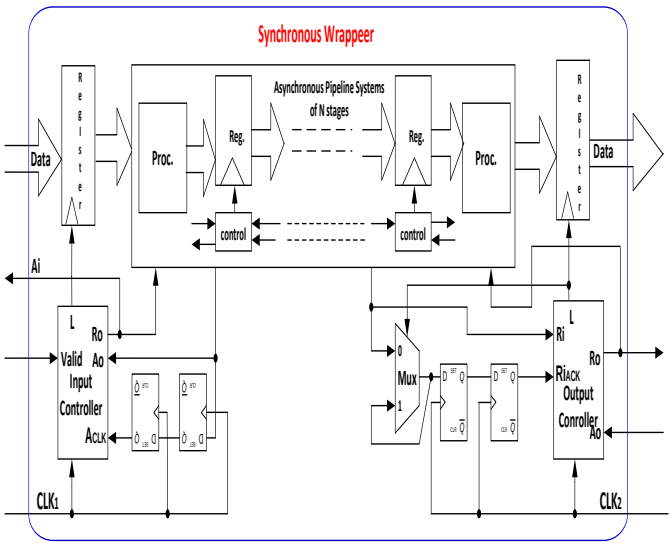

Fig. 4. Proposed synchronous wrapper for asynchronous pipeline module.

II. Asynchronous Pipeline

Asynchronous pipelines have several properties that potentially benefit the circuit design. The most promising property is the possibility of selecting the pipeline stage only in the presence of valid data [6,19]. The same techniques and tools used in the synchronous pipeline project can be used in asynchronous design, through a simple desynchronization. The interaction of synchronous wrapper with other synchronous modules occurs as shown in Fig. 3. The locally asynchronous pipeline of synchronous wrapper is implemented in the same architecture proposed in [19]. Figures 5a and 5b show the control description that was specified in the STG of [20] and the logic circuit of the control.

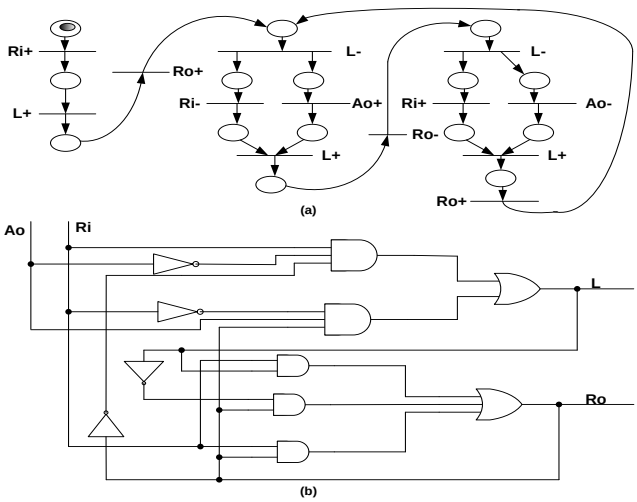

Fig. 5. Control of [19]: a) STG description; b) logic circuit.

III. Input Port Controller

The input port of the proposed wrapper works in a "synchronous to asynchronous" environment. It is synchronized with the synchronous modules when it accepts the clock signal. Then, it interacts with the asynchronous pipeline in the two-phase protocol. In this protocol, the processing "request" occurs on both edges of the *Ro* signal, while the "acknowledge" of *Ao* signal needs a synchronization [6]. Figure 6 shows a general layout of the input port, which consists of two flip-flops for synchronization and an input port controller. The input controller was specified in XBM (extended burst-mode specification) of [21] (Fig. 7). It is composed by the following signals: *Valid* (data validity), *Ao* (acknowledge output), *CLK* (clock signal), *L* (load) and *Ro* (output request).

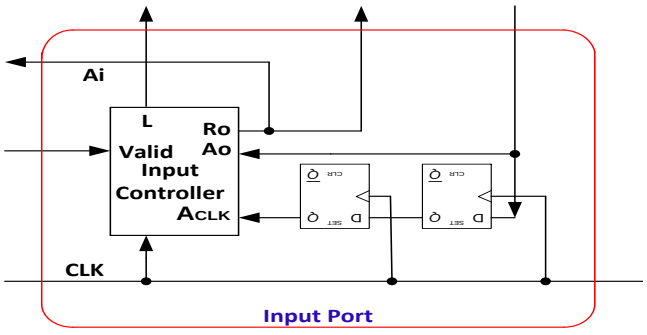

Fig. 6. Scheme of input port controller.

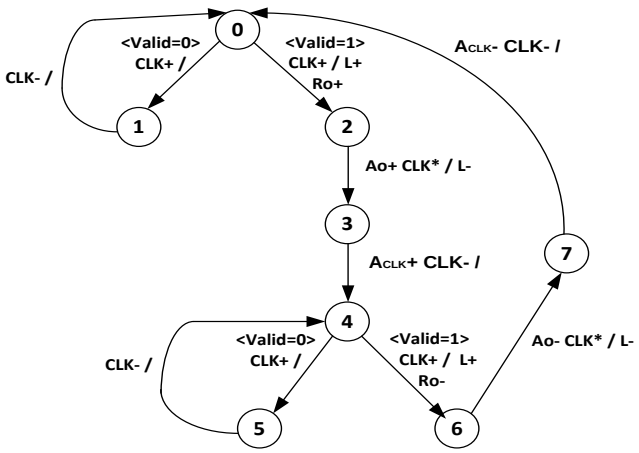

Fig. 7. Controller of input: XBM specification.

IV. Output Port Controller

The output port controller of the proposed wrapper works in an "asynchronous to synchronous" environment, contrary to what was previously seen. The port receives the clock signal and therefore is synchronized with the synchronous modules. The output port controller interacts with the asynchronous pipeline in the two-phase protocol, where the processing "request" occurs at both edges of the *Ri* signal and the "request" needs synchronization. Figure 8 shows the general layout of the output port, which consists of two flip-flops for synchronization, a 2x1 mux and an output port. The output controller was specified in XBM (Fig. 9), being composed by the following signals: *Ri* (request input), *Ao* (acknowledge output), *CLK* (signal of clock), *L* (load) and *Ro* (output request).

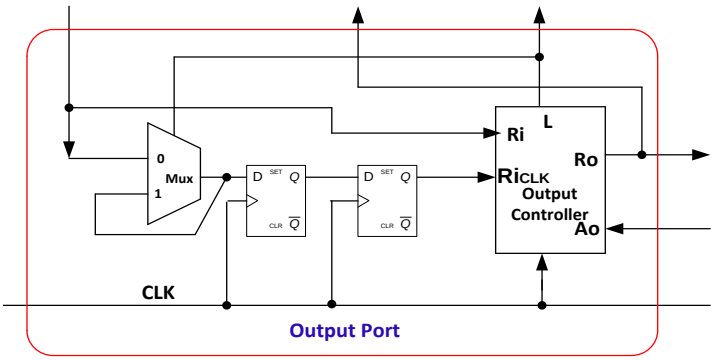

Fig. 8. Scheme of output port controller.

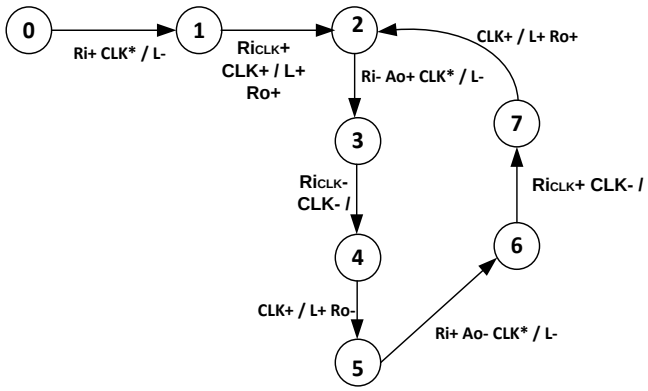

Fig. 9. Controller of output: XBM specification.

V. Case Study

In order to illustrate the use of the proposed interface, an asynchronous pipeline version of a 5th order FIR filter was designed, based on (1).

$$Y[t] = \sum_{i=0}^{N-1} H[i] \times X[t-1] \qquad (1)$$

where, H_0, H_1,...,$H_{(N-1)}$ are constant weights. The FIR filter, used in this paper, was already designed in [19] and was obtained from the List Scheduling Algorithm with three resource constraints: two multipliers and one adder [22]. The method of [19] generates a pipeline data-path with six stages of bundled-data and matched delay type. Figure 10 shows the pipeline data-path of the filter FIR, where the registers are implemented with D flip-flops. The asynchronous pipeline control requires six controls and five delay elements.

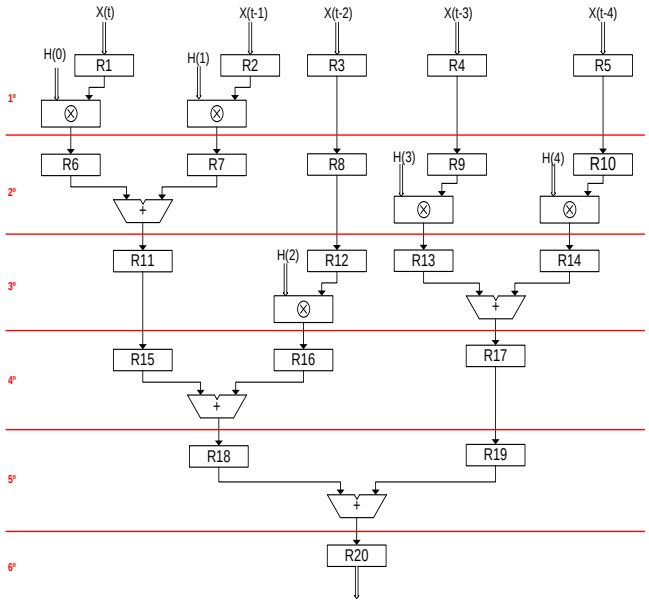

Fig. 10. Pipeline data-path of the filter FIR of [19].

VI. Simulations & Results

The simulations and designs of the synchronous wrapper with 5th order FIR filter and the corresponding synchronous version were performed in Quartus II software, version 9.1 [23], considering Altera STRATIX II (EP2S15F484C3) as target device.

A. *Simulation & results of controllers of input and output*

Figures 11 and 12 show hazard-free simulations for the input and output controllers. The waveforms are exactly as expected to the SAPS (Synchronous to Asynchronous Pipeline to Synchronous) design. The input port controller is responsible for the communication of the synchronous module to the asynchronous pipeline and is placed before the first stage of the asynchronous pipeline. On the other hand, the output port controller is responsible for the communication of the asynchronous pipeline to synchronous modules and is located after the last stage of the asynchronous pipeline. Both controllers were synthesized in 3D tool [21]. The input and output ports allow the interacting of the synchronous wrapper with other synchronous modules in a frequency up to *500MHZ*.

Table I summarizes the achieved results for the input and output ports concerning to area (LUTs + FFs), time of load and time of request (Ro).

Fig. 11. Simulation: input port controller of wrapper.

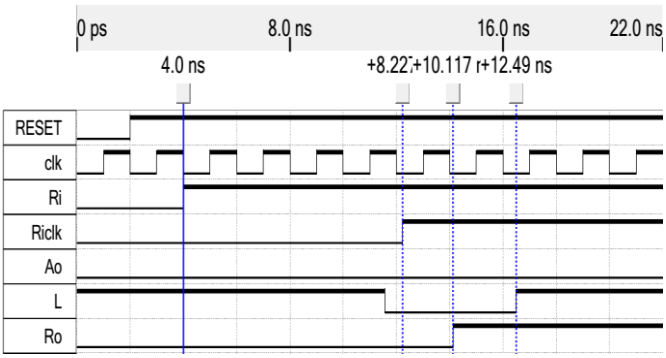

Fig. 12. Simulation: output port controller of wrapper.

TABLE I Results: ports of input & output

Port	Time of Load	Time of Request (Ro)	Number of LUTs	Number of FFs
Input Port	10.88ns	10.03ns	10	2
Output Port	12.49ns	10.12ns	7	2

B. *Simulation & results of FIR filter*

Figure 13 shows the hazard-free simulations of the FIR filter, exactly as expected. Table II presents the results of the 5th order FIR digital filter, considering latency, power dissipation and area. Results allow comparing the proposed interface with the one with a synchronous pipeline. The used synchronous pipeline does not incorporate the necessary additional blocks, as the controllers. It is simply the data-path shown in Fig. 10 with the use of the clock, what leads to a much better performance than the real one. Even though, our full system presents better results. The proposed interface presented a reduction of 4% in the power dissipation, an increase of 35% in the number of macrocells (LUTs + FFs) and an increase of 0.6% in the latency time. The synchronous wrapper provided an increase of 200% in global clock rate.

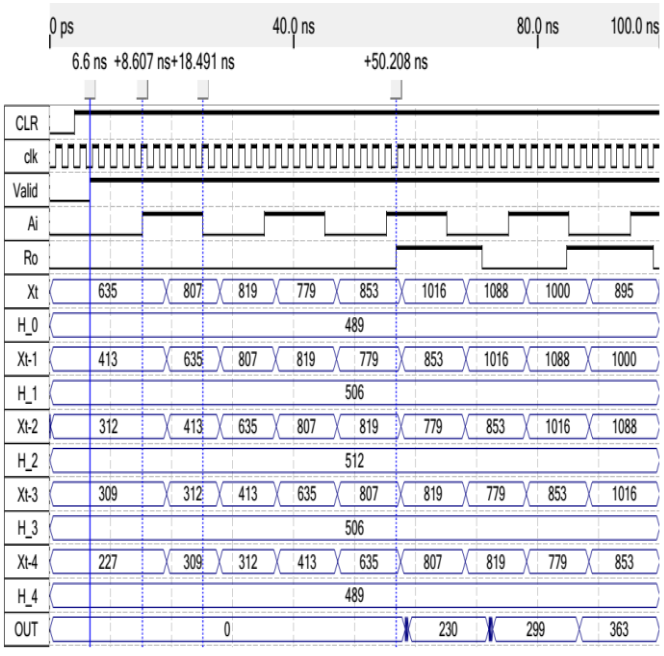

Fig. 13. Simulation: synchronous wrapper with FIR Filter.

TABLE II RESULTS: PIPELINE DIGITAL FIR FILTER

		Time of Latency	Power Dissipation	Macrocells	
				Number LUTS	Number Flip-Flops
Filter FIR	Synchronous f_{MAX}=166 MHZ	**49.903ns**	418.95mw	**55**	**280**
	Proposal Wrapper	50.208ns	**402.54mw**	137	316

C. *Analysis of synchronous wrapper*

The implementation of the proposed synchronous wrapper contains an asynchronous pipeline module that provides some alternatives for the designing of a global, or multiple, clocks ASIC system:

- The synchronous wrapper reduces the power of the global clock signal of the ASIC-SoC system because, besides not acting in the registers of the asynchronous modules, it also reduces the number of buffers along the circuit.
- The synchronous wrapper reduces the clock skew problem, due to the reduction of buffers and the reduction of the length of the clock lines.
- The synchronous wrapper allows increasing the overall clock rate of the SoC system because the synchronous modules with lower clock rate are implemented as asynchronous ones.
- Analyzing the case study, it was observed a negligible increase in the latency time. Although existing an increasing in area, this requirement is not significant to the current integration level.

VII. CONCLUSION

Complex digital systems can be designed in GSLA style, presenting good results and overcoming some existing drawbacks. This style allows taking advantage of both synchronous and asynchronous paradigms. It is possible to analyze the performance, power reduction and design facility of each module and then deciding in which paradigm the module will be synthesized.

In this paper, a dedicated interface for FPGAs is proposed. It comprises an internally asynchronous pipeline, communicating externally in a synchronous mode, and allowing a "synchronous-asynchronous-synchronous" communication. Through a case study, it is shown that the results show to be promising, achieving power reduction even when compared with an extremely simple design of synchronous pipeline. In the case study, the FIR filter operates with maximum frequency of 166 MHz, representing the global clock rate, if it was the lowest rate module. The synchronous wrapper provided a 200% increase in the overall clock rate, because it operates at rates up to 500MHz. Furthermore, the interface allows an interaction with synchronous modules up to 500 MHz, showing good results and a high potential for practical implementation.

REFERENCES

[1] G. DE Micheli, "An Outlook on Design Technologies for Future Integrated Systems," IEEE Trans. on CAD of Integrated Circuits and Systems, vol28, no.6, pp. 777-789, June 2009.

[2] K. D. Muller-Glaser, et. al. "Multiparadigm Modeling in Embedded Systems Design", IEEE Trans. on Control Systems Technology, vol. 12, no. 2, March 2004.

[3] J. Cortadella, A. Kondratyev, L. Lavagno, and C. Sotiriou, "Coping with the variability of combinational logic delays," *ICCD*, pages 505–508, 2004.

[4] J. J. Rodriguez, et. Al., "Features, Design Tools, and Applications Domains of FPGAs", *IEEE Trans. on Industrial Electronics*, vol. 54, No. 4, pp.1810-1823, August 2007.

[5] M. Rubini, and C. Rajasekaran, "Design of high performance system-on-chips using Field Programmable Gate Arrays (FPGA)," 2014 International Conference on Communications and Signal Processing (ICCSP), pp.358-362, 2014.

[6] C. J., Myers, "*Asynchronous Circuit Design*", Wiley & Sons, Inc., 2004, 2ª edition.

[7] D. M. Chapiro, *Globally-Asynchronous Locally-Synchronous Systems*, PhD thesis, Stanford University, October 1984.

[8] A. E. Sjogren and C. J. Myers, "Interfacing Synchronous and Asynchronous module Within a High-Speed pipeline," IEEE Trans. on Very Large Scale Integration (VLSI) Systems, Volume: 8 , Nro. 5, pp. 573-583, 2000.

[9] K. Stevens, S. Rotem, R. Ginosar, P. Beerel, C. J. Myers, K. Yun, R. Koi, C. Dike, M. Roncken, "An asynchronous instruction length decoder," *IEEE Journal of Solid-State Circuits*, Volume: 36 , Nro. 2, pp.217-228, 2001.

[10] M. Singh, J. A. Tierno, A. Rylyakov, S. Rylov, and S. M. Nowick, "An adaptively-pipelined mixed synchronous-asynchronous digital FIR filter chip operating at 1.3 gigahertz," in *Proc.* Eighth International Symposium on Asynchronous Circuits and Systems, pp. 84-95, April 2002.

[11] M. Singh and S. M. Nowick, "The Design of High-Performance Dynamic Asynchronous Pipelines High-Capacity Style", *IEEE Trans. on VLSI Systems*, vol.15, no.11, ´pp.1270-1283,November, 2007.

[12] S. Moore, G. Taylor, R.t Mullins, P. Robinson, " Point to Point GALS Interconnect," Proc. Eighth International Symposium on Asynchronous Circuits and Systems, pp. 69-75, 2002.

[13] S. E. Schuster and P. W. Cook, "Low-Power Synchronous-to-Asynchronous-to-Synchronous Interlocked Pipelined CMOS Circuits Operating at 3.3–4.5 GHz," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 4, pp. 622-630, April 2003.

[14] H. M. Jacobson, "*Interlocked Synchronous Pipelines*," PhD thesis, University of Utah, 2004.

[15] T. Chelcea and S. M. Nowick, "Robust interfaces for mixed-timing system," *IEEE TVLSI*, vol. 12, no. 8, pp.857-873, Aug 2004.

[16] F. K. Lodhi, S. R. Hasan, O. Hasan and F. Awwad, "Low Power Soft Error Tolerant Macro Synchronous Micro Asynchronous (MSMA) Pipeline," Proc. IEEE Computer Society Annual Symposium on VLSI, 2014.

[17] P. W. Cook and S. E. Schuster, "*Synchronous to Asynchronous to Synchronous Interface*," Patent number 6,848,060 B2, January 25, 2015.

[18] W. Ho, K. Chong, B. Gwee and J. S. Chang, "Low Delay-Variation Sub-/Near-Threshold Asynchronous-to-Synchronous Interface Controller for GALS Network-on-Chips," IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 5-8, 2014.

[19] D. L. Oliveira, et al., "Using FPGAs to Implement Asynchronous Pipeline," 5th IEEE Latin American Symposium on Circuits and Systems, Santiago, Chile, 2014.

[20] T. –A. Chu, "*Synthesis of Self-Timed VLSI Circuits from Graph-Theory Specifications*," PhD. Thesis, June, 1987, Dep. Of EECS, MIT.

[21] K. Y. Yun e D. L. Dill, "Automatic Synthesis of Extended Burst-Mode Circuits: Part I (Specification and Hazard-.Free Implementation) and Part II (Automatic Synthesis)," *IEEE Trans. on CAD of Integrated Circuit and System*s, Vol. 18:2, pp. 101-132, Feb. 1999.

[22] D. D. Gajski, "*Principles of Digital Design*," Prentice Hall, 1997.

[23] Altera Corporation, 2016, www.altera.com.