ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Novel Implementation of Asynchronous Pipelining using FIR filters on FPGA

N.Krishnaveni
M.Tech VLSI,
Department of ECE,
CMR Institute of Technology.

K.Narmada
Assistant Professor,
Department of ECE,
CMR Institute of Technology.

MD.Shabhaz Khan
Associate Professor,
Department of ECE,
CMR Institute of Technology.

Abstract :

The asynchronous paradigm has interesting features due to the lack of the clock signal and it is another option for the project of digital systems. This paradigm has several design styles, where the micropipeline style is the most suitable one for FPGA platforms, due to the simplicity of its control. In this paper, we propose a pipeline architecture to implement asynchronous digital systems, in bundled-data micropipeline style, considering FPGAs as target devices. Through a case study, we show that the proposed architecture presents a 29% decrease in latency time and a 13% increase in throughput, compared with the state of the art architecture MOUSETRAP.

I. INTRODUCTION:

FPGAs devices (Field Programmable Gate Array) have become a very popular way to develop and implement digital circuits due to their cost and design time. High performance FPGAs are implemented in Deep-Sub-Micron MOS technology (DSM-MOS). In this technology, the delay on the lines should be considered and may be higher than the delay of the logic gates [1], [2]. Digital systems are traditionally designed in synchronous paradigm, i.e. they use a global clock signal to synchronize their operations.

They are quite popular due to their simplicity of design and availability of commercial CAD tools for automatic synthesis. In DSM-MOS technology, a clock signal requires attention due to its noise generation, electromagnetic interference and power consumption. Besides these factors, the distribution of the clock signal along the chip is a task with increasing complexity because of the clock skew problem, which drops the system performance. The overhead caused by the clock signal can reach 130% in a VLSI (Very Large Scale Integration) implementation [3] and worsens when FPGAs are employed.

Considering the design style of asynchronous systems, it is easier to implement the micropipeline style in commercial FPGAs due to the simplicity of its control [4], [5]. This is important because of the difficulty to achieve hazard-free Considering the design style of asynchronous systems, it is easier to implement the micropipeline style in commercial FPGAs due to the simplicity of its control [4], [5].

This is important because of the difficulty to achieve hazard-free [7], [8], [9], [10], [11], [12], [13]. However, most of these pipeline architectures are focused on VLSI implementations, employing full-custom control [8], [9], [10]. FPGA oriented pipelines have been proposed [12], [13], but one uses a complex delay mechanisms [12] and the other does not obey the fundamental-mode [13].

The pipeline architecture known as MOUSETRAP, proposed in [11], has a good performance, it is based on logic gates and can be implemented on FPGAs (see Fig. 1).In this paper, we propose a FPGAs oriented architecture for micropipeline design style (see Fig. 2).

The proposed linear pipeline architecture employs flip-flops as registers, due to the large availability of these elements in FPGAs. The control is based on logic gates and it is mapped into two LUTs (Look-Up Table).

The new pipeline architecture has a better performance, whencompared to the MOUSETRAP architecture implemented on FPGA, because it has a smaller latency and a greater throughput. A digital FIR filter of fourth order shows the efficiency of our architecture on a FPGA platform.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

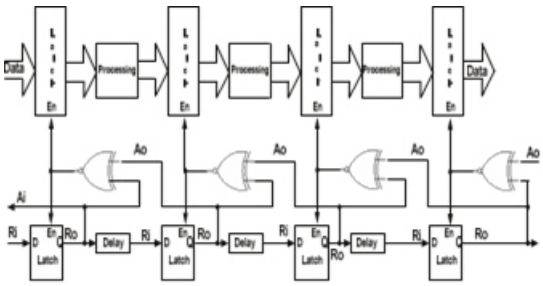


Fig. 1. MOUSETRAP linear micropipeline [11]

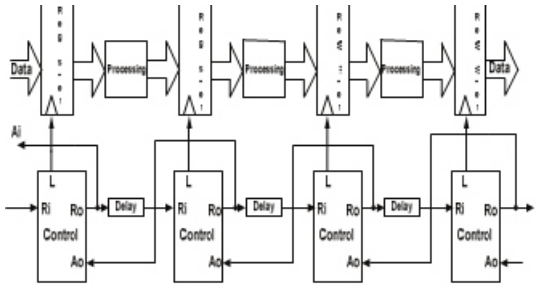
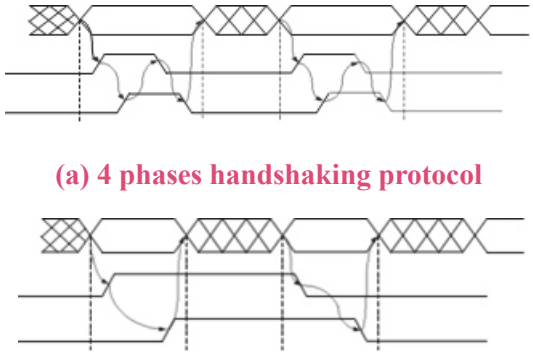


Fig. 2. Proposed linear micropipeline architecture.

II. MICROPIPELINE PROJECT EMPLOYING FPGAS:

Programmable devices, such as FPGAs, are designed for synchronous designs [2]. Efforts for prototyping asynchronous design in commercial FPGAs [14], [15] and academic FPGAs [16], [17] been reported recently. The problems in implementing asynchronous systems in commercial FPGAs are related to the control. They are:a) Process of mapping risk-free boolean functions in logic blocks (macrocells). The commercial tools used for decomposition and mapping boolean functions in LUTs are not prepared to meet the requirements of logical hazard. This may cause a circuit malfunction, if manual intervention to fix the problem is not performed. The mapping function must satisfy the decomposition requirements proposed in Sigel et al. [18]. b) Internal routing process among macrocells can introduce significant delays. These delays can result in essential hazard and lead to the circuit malfunction [4], [5]. The circuit delay model defines how to solve the problem of essential hazard: the insertion of delay elements in the feedback lines or the employment of macrocells that satisfy the isochronic fork condition [4], [5]. The micropipeline architecture can be linear or nonlinear. [19]. In this paper we focus only on the linear architecture.

The micropipeline style has as main feature, the simplification of the pipeline control, an important characteristic for asynchronous systems implemented in FPGAs. The control is either distributed between stages or centralized, and it is responsible for the communication between the pipeline stages. In a FPGA platform, the control must be distributed in order to avoid hazard problems, as previously mentioned. The pipeline communication employs the handshake protocol with Request and Acknowledge signals [4], [5]. The communication between the stages can be performed in twodifferent protocols: 4-phase or 2-phase. Fig. 3 shows the behavior of these protocols.



(a) 4 phases handshaking protocol

Fig. 3. Handshake protocol: a) 4-phase; b) 2-phases

The linear micropipeline design has two variants. In the first one, it is used only components of the synchronous paradigm (single-rail) and delay elements between stages. The delay is defined considering the critical path of each stage. The bundled-data implementation is one of the data encoding schemes used in asynchronous circuits. It represents N bits of data with N+2 lines, called “bundled”, where the two additional lines are the handshake signals, request and acknowledge. This architecture is called “micropipeline bundled-data”.The second architecture employs dual rail components [4], [5]. The components are synthesized using Delay Insensitive(DI) codes or dual-rail. In the case of dual-rail code, each Fig. 6. Controller’s logic circuit. signal is encoded with two bits. Four cases would be possible for a signal: a0a1=00 (null); a0a1=01 (1); a0a1=10 (0); and a0a1=11 (never occurs). The design employing this code generates a signal of “end of operation” without the need of delay element, and it can be implemented with a simple circuit. The dual-rail linear micropipeline architecture can be implemented in FPGA platforms, but there is a significant increase in the number of used macrocells, that compromises their performance and power consumption. On the other hand, some advantages need to be considered:

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

this architecture doesnot need delayelements and timing analysis, moreover, it ismore robust. The proposed micropipeline architecture operates in 2- phase handshake protocol and, therefore, the processing request occurs on both edges of this signal. Fig. 4 shows the general layout of a stage, which consists of a flip-flop based register and control. The control was specified in STG (signal transition graph) [20] (see Fig. 5). It is composed by the signals: Ri (request input), Ao (acknowledge output), Ai (acknowledge input), L (load) and Ro (output request). Fig. 6 shows the circuit of the logic control.

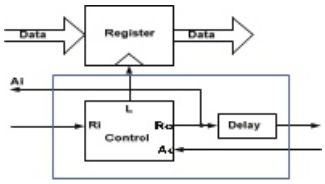


Fig. 4. Linear micropipeline target for FPGA.

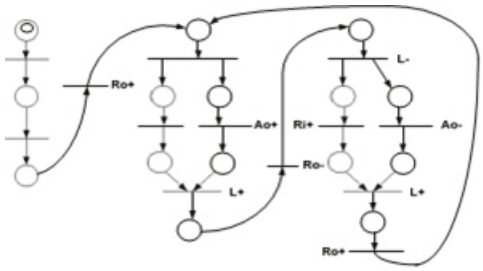


Fig. 5. Controller's signal transition graph

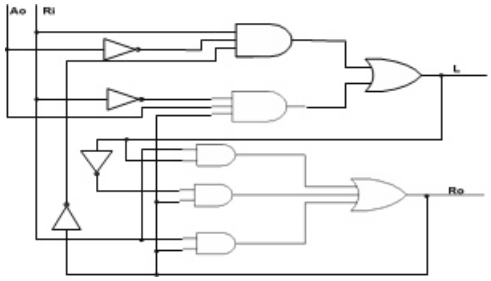


Fig. 6. Controller's logic circuit.

IV. SYNTHESIS OF PIPELINE SYSTEMS:

The procedure used to synthesize bundled-data micropipeline systems attempts to use components and synthesis tools of the synchronous paradigm. The behavioral synthesis starts from the Control Data Flow Graph (CDFG), which represents the operations and their data dependency. The method can be divided into five steps: 1) Generation of the scheduled DFG employing the list scheduling algorithm [21]. 2) Based on the scheduled DFG, generation of the pipeline data-path employing single-rail components and the pipeline synthesis according to [21]. This step generates the synchronous pipeline. (see Fig. 7). 3) Execution of the desynchronization process; detection of the critical path of each pipeline step and calculation ofthe respective delay element. 4) Change the clock signal by the proposed asynchronous pipeline controller (see Fig. 8).5) Synthesize the asynchronous pipeline generated in step 4.

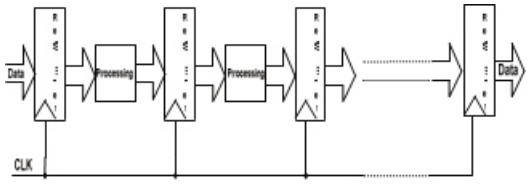


Fig. 8. Asynchronous pipeline architecture.

In order to illustrate the use of the proposed micropipeline architecture, an asynchronous version of a 4th order FIR filter was designed, according to the equation:

$$y[n] = \sum_{i=0} h_i \cdot x[n-i] \qquad (1)$$

where, y[n] and x[n] are the output and input signals samples, and h(i) are the FIR filter coefficients.Fig. 9 shows the first step, the scheduled GFD of the filter, obtained from the List Scheduling algorithm with three resource constraints: two multipliers and one adder. The second step of the proposed method generates a data-path pipeline with five stages. It was obtained by the behavioral synthesis [21], which defines the synchronous pipeline. Fig. 10 shows the pipeline data-path of the filter. The steps 3 and 4 make the desynchronization of the pipeline, generating the proposed pipeline, requiring six controls and six delay elements.

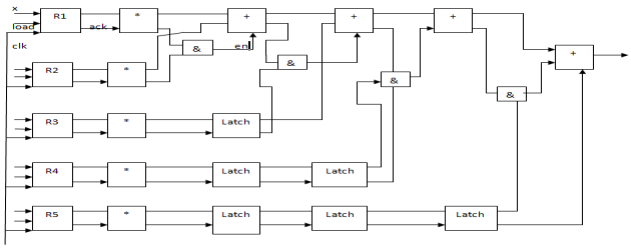


Fig. 9. Scheduled GFD of the filter.

VI. EXPERIMENTAL RESULTS:

A comparison was made with the MOUSETRAP pipeline architecture in order to evaluate the performance of the proposed architecture.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

The simulations were performed employing the Xilinx software, version 14.2 , considering an Spartan 6 as target device. Fig. 11 and 12 show the simulations of the mousetrap control and the stages of the FIR filter. Fig. 13 and 14 show the simulations of the proposed pipeline: control and FIR stages.

ARRAY MULTIPLIER asynchronous pipeline

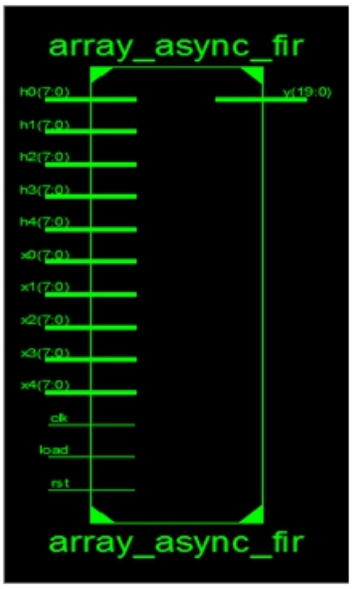
Area report

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	464	768	60%
Number of Slice Flip Flops	251	1536	16%
Number of 4 input LUTs	808	1536	52%
Number of bonded IOBs	103	124	83%
Number of GCLKs	3	8	37%

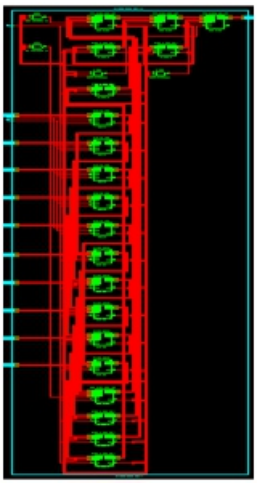
Delay report

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 17.486ns (Maximum Frequency: 57.189MHz)
   Minimum input arrival time before clock: 30.149ns
   Maximum output required time after clock: 6.141ns
```

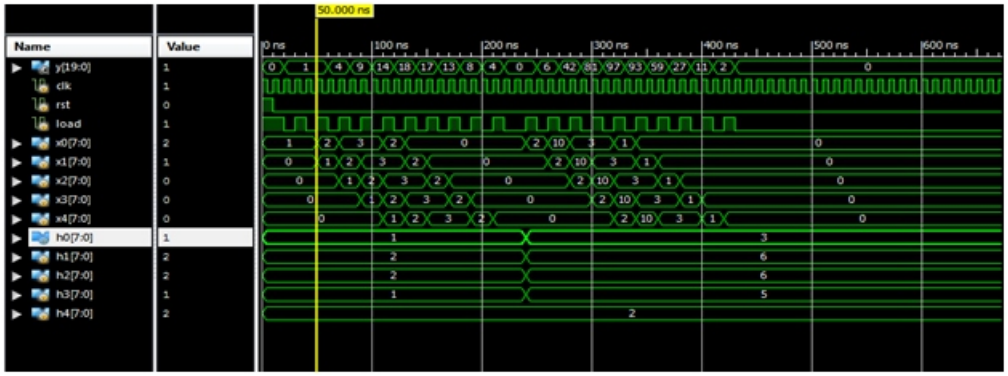


Block diagram for Array asynchronous pipeline



RTL schematic

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal



WAVEFORM

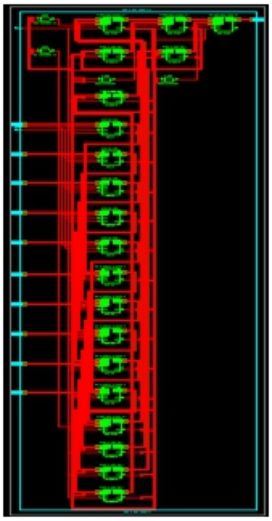
BOOTH MUL USING ASYNCHRONOUS PIPELINE

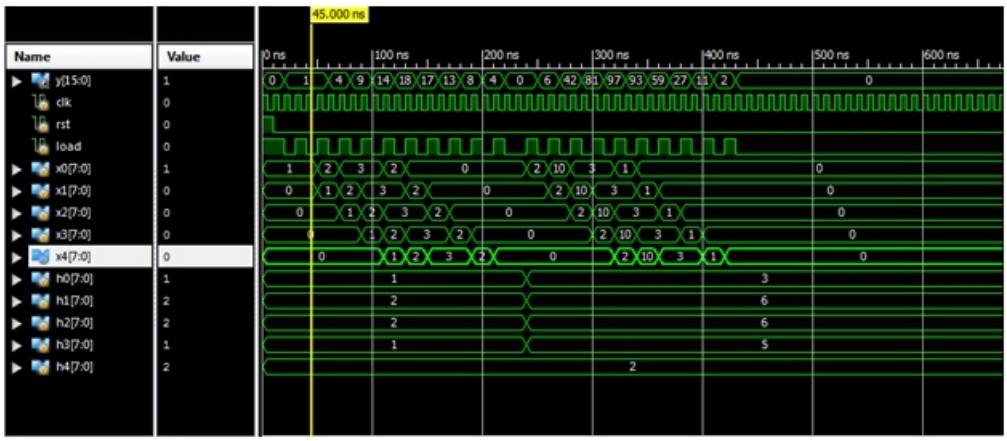
Area report

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	416	768	54%
Number of Slice Flip Flops	241	1536	15%
Number of 4 input LUTs	749	1536	48%
Number of bonded IOBs	99	124	79%
Number of GCLKs	3	8	37%

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 14.994ns (Maximum Frequency: 66.693MHz)
   Minimum input arrival time before clock: 18.741ns
   Maximum output required time after clock: 6.141ns
```





Delay report

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

VII. CONCLUSION:

The linear micropipeline style of the asynchronous paradigm is advantageous because the performance is increased and the control is simplified. In this paper, we proposed a new pipeline architecture oriented for FPGAs implementations. The control employs only two LUTs, and the registers used are based on flip-flops, allowing a better distribution of macrocells in a FPGA. The implementation of the proposed FPGA control is free of essential hazard. Through a case study, we show that our architecture has a better performance than the MOUSETRAP architecture, considering the FPGA implementation.

REFERENCES:

[1]D. Goldhaber-Gordon, et al., "Overview of Nanoelectronic Devices," Proc. of the IEEE, vol. 85, No. 4, pp.521-540, April 1997.
[2]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.
[3]J. Cortadella, et al., "Coping with the variability of combinational logic delays," ICCD, pages 505–508, 2004.
[4] C. J., Myers, "Asynchronous Circuit Design", Wiley & Sons, Inc., 2004, 2a edition.
[5] J. Sparsø e S. Furber, "Principles of Asynchronous Circuits Design", Kluwer Academic Publishers, 2001.
[6]I. E. Sutherland, "Micropipelines", Communication of the ACM, vol. 32, No.6, pp.720-738, June, 1989.
[7]S. B. Furber et al., "Four-Phase Micropipeline Latch Control Circuits," IEEE Trans. on VLSI Systems, vol.4, no. 2, pp.247-253, June, 1996.
[8]G. S. Taylor and G. M. Blair, "Reduced Complexity Two-Phase Micropipeline Latch Controller," IEEE Journal of Solid-State Circuits, vol. 33, Nro. 10, pp.1590-1593, October, 1998.
[9]M. Singh and S. M. Nowick, "The Design of High-Performance Dynamic Asynchronous Pipelines: Lookahead Style," IEEE Trans. on VLSI Systems, vol.15, no. 11, pp.1256-1269, November, 2007.
[10] 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.
[11] M. Singh and S. M. Nowick, "MOUSETRAP: High-Speed Transition-Signaling Asynchronous Pipelines", IEEE Trans. on VLSI Systems, vol.15, no. 6, pp.684-698, June, 2007.
[12] T.N. Prabakar G. Lakshminarayanan, K. K. Anilkumar, "FPGA Based Asynchronous Pipelined Multiplier with Intelligent Delay Controller," International SOC Design Conference, 304-309, 2008.
[13] D. L. Oliveira, et al., "An Architecture for High Performance Micropipeline Systems directed to FPGAs," XV SIGE, (in Portuguese).
[14] E. Brunvand, "Using FPGAs to Implement Self-Timed Systems", Journal of VLSI Signal Processing, Special issue on field programmable logic, vol.6(2), pp.173-190, August, 1993.
[15] M. Tranchero and L. M. Ryneri, "Implementation of Self-Timed Circuits onto FPGAs Using Commercial Tools", 11th Euromicro Conf. on Digital System Design Architectures, Methods and Tools, pp.373-380, 2008.
[16] R. Payne, "Asynchronous FPGA architectures," IEE Proc. Comp. Digit. Tech., vol.143, no.5, September, pp.282-286, 1996.
[17] N. Huot, et. al., "FPGA architecture for multi-style asynchronous logic," Proc. Of the Design, Automation and Test in Europe Conference and Exhibition, 2005.
[18] P. Sigel, G. De Michele and D. Dill, "Decomposition methods for library binding of speed-independent," Proc. Int. Conf. Computer-Aided Design, pp.558-565, 1994.
[19] R. O. Ozdag , et al., "High-Speed Non-Linear Asynchronous Pipelines," Proceedings of the 2002 Design, Automation and Test in Europe Conference and Exhibition (DATE.02), 2002.
[20] T. –A. Chu, "Synthesis of Self-Timed VLSI Circuits from Graph-Theory Specifications," PhD. Thesis, June, 1987, Dep. Of EECS, MIT.
[21] D. D. Gajski, "Principles of Digital Design," Prentice Hall, 1997. [22] Altera Corporation, 20013, www.altera.com.