Rapid prototyping of a Self-Timed ALU with FPGAs

[1]Ortega-Cisneros S., [1]Raygoza-Panduro J.J., [2]Suardíaz Muro J., [1]Boemo E.
[1]*Escuela Politécnica Superior, Universidad Autónoma de Madrid, España*
[2]*Escuela Técnica Superior de Ingenieros Industriales, Universidad de Cartagena*
susana.ortega@uam.es, jjraygoza@uicm.net

Abstract

This article presents the design and implementation of a Self-Timed Arithmetic Logic Unit (ALU) that has been developed as part of an asynchronous microprocessor. This displays an inherent operational characteristic of low consumption, owing to the synchronization signals that stop when the execution of an operation finishes (stoppable clock); that is to say, the dynamic consumption is zero, while it is not required again by an external request signal.

It demonstrates the methodology of design of the Self-Timed controls which synchronize the data transfer, as well as the characterization of delay macros designed in FPGA editor for the adjustment of ALU processing times. It also summarizes the results of the implementation for a FPGA virtex II, as well as the parameters of area, distribution of tracks, delay, latency, consumption and fan-out.

1. Introduction

The design of non-synchronous digital systems constitutes an alternative for synchronizing large circuits. Therefore the methodology of self-timed (ST) design has advanced in the recent years. Among the advantages we can mention is its inherent operation in stoppable-clock mode, the absence of consumption peaks and its immunity to the skew of the clock. In a synchronous circuit the transmission or data processing is controlled globally by one or more phases of the clock. Whereas in a ST system, the data transfer is controlled by two signals: "request and acknowledge" as is normal in any asynchronous system [1]. The definition of the control signal format gives rise to two types of synchronization: protocol of 2 phases [2] and protocol of 4 phases [3,4]. This article presents the implementation of an ALU using the protocol of 4 phases. This has been selected in place of 2 phases, due to the robustness of the technique, simplicity of implementation of the transmission blocks and the minimum use of the resources of the FPGA device. [5,6].

At the present time the development of ST circuits has been centered on full-custom or cell-based prototypes, although the FPGAs are oriented towards the efficient implementation of synchronous circuits, at the present they constitute the only option available for fast prototypes and for the low cost of self-timed circuits.

2. Description of a ST ALU

The ST Arithmetical Logical Unit has been developed as part of the asynchronous implementation of a microprocessor. The asynchronous circuit is composed of 4 main modules, as shown in the figure 1:

1. Arithmetical and Logical Unit
2. Instructions decoder
3. Asynchronous control
4. ST 4 phases pipeline

The Arithmetical Logical Unit is a combinatorial device composed of 3 main modules as can be seen in figure 1. The Module (a) or instruction block has a feeding signal of 16 bits, which enters directly from the outside to one of the two instruction entrances. The other entrance is feedback from the exit of the accumulator (module c).

The ALU has 4 arithmetical instructions, 6 logics, 1 of comparison, 2 of register transference and 2 of input-output.

The selection of the functions is made by means of the 15 signals “Io” up to “I14”, that activate the 15 channels of the multiplexor to illustrate the result of the operations (module b), which are related to the entrances and allow the passage of one of the logic, arithmetic or input-output functions.

Proceedings of the 2005 International Conference on Reconfigurable Computing and FPGAs (ReConFig 2005)
0-7695-2456-7/05-$20.00 © 2005 **IEEE**

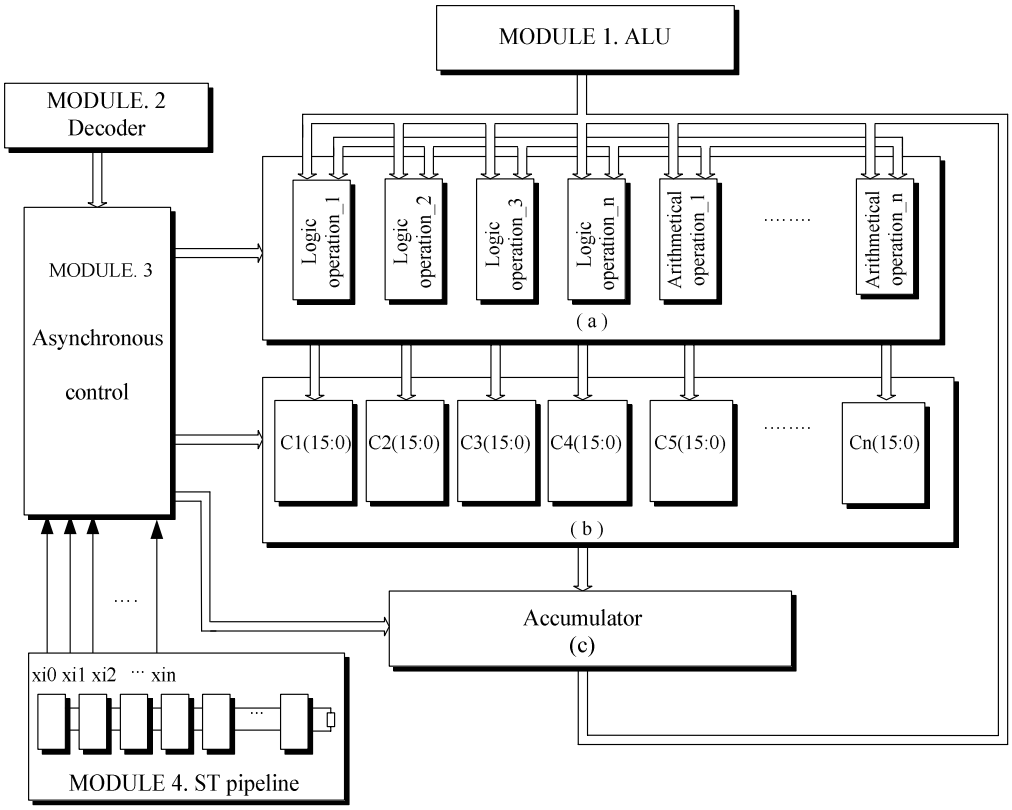


Figure 1. ST Arithmetical Logical Unit

The general module has 5 codifications for 15 instructions. Table 1 shows the selection code, the decoding of the operations (deco) and the occupation of these in the FPGA.

3. Asynchronous Control

The instructions are classified into 4 types of different operations, according to the number of activation pulses that are required for their execution. These are shown in the diagram of figure 2 and constitute the asynchronous control. The operations of type 1 require 4 pulses at the entrance that come from the ST 4 phase pipeline module to activate the ALU.

The operations of type 2, require 2 pulses at the entrance, the operations of type 3 require 5 pulses and finally the operations type 4 require 9 pulses to codify 4 activation signals from the ALU. Some control lines are concentrated in an exit circuit allowing the accumulator to capture the data correctly. The line of "total test" of the circuit in figure 2, generates a pulse whenever an instruction is made, which is connected to an operations counter.

Instruction	MUX line	Selection	Deco	Occupation			
				Sl	LUT	Reg.	Gates
LDA	I2	00001	0001	-	-	-	-
ADD	I1	00010	0002	9	17	0	186
ROT_D	I3	00011	0004	0	0	16	131
ROT_I	I4	00100	0008	0	0	16	131
COMPL	I5	00101	0010	22	42	0	369
DES_D	I6	00110	0020	0	0	16	123
LDA, X	I9	00111	0040	0	0	16	131
INC, A	I8	01000	0080	13	21	0	126
COMP	I7	01001	0100	0	0	16	131
LDA, Y	I10	01010	0200	0	0	16	131
AND	I11	01011	0400	16	16	0	96
OR	I12	01100	0800	16	16	0	96
PTO_SAL	-	01101	1000	0	0	16	131
RESTA	I14	01110	2000	9	17	0	189
MUL	I13	01111	4000	0	0	16	4,134

Table 1. ALU Instructions

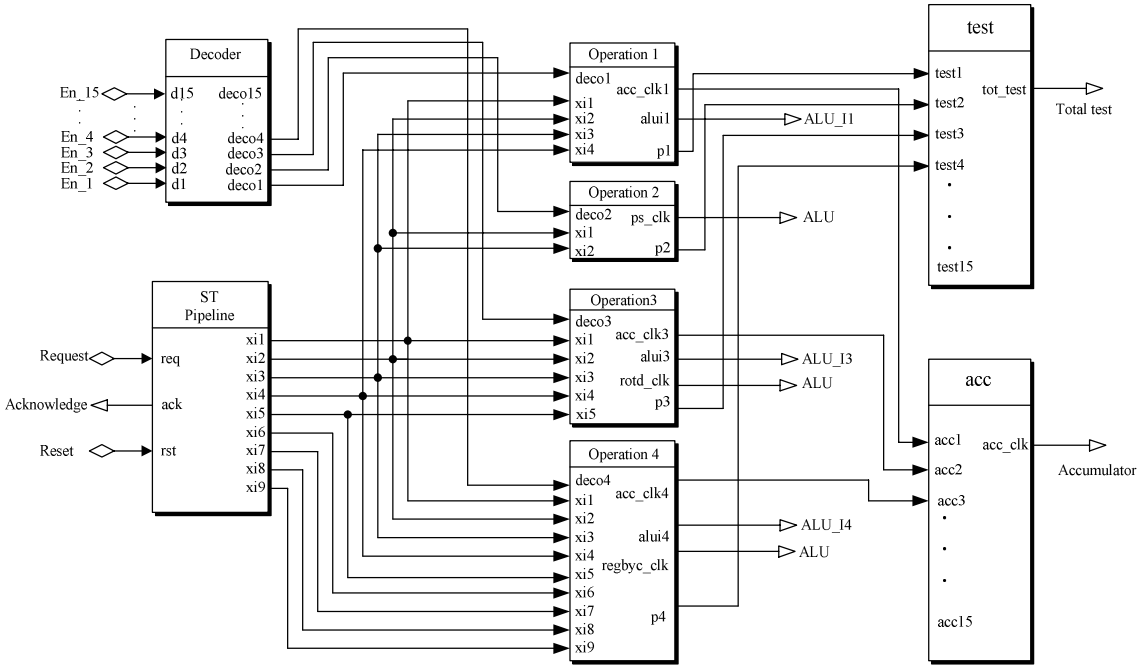


Figure 2. Asynchronous control

The decoding of operation 1 is shown in figure 3, it requires several logic gates to make the transformation of the control signals "xi1" to "xi4" in order to activate the signals of the multiplexer and the accumulator; the elements comp_1 to comp_3 are controlled by the signal "deco1" that comes from the instructions decoder, permitting the transmission of the signals as long as "deco" signal is actived.

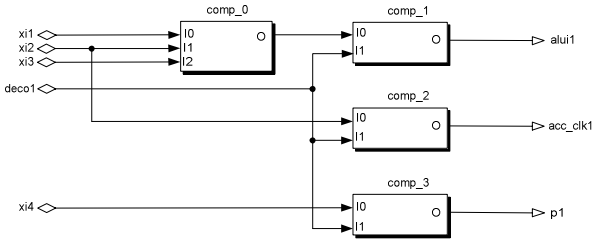


Figure 3. Logic operation 1

The operation 2 requires 2 logical elements to transform the pulses from 2 signals from the ST control pipeline. One of these is required to activate the capture of the register and the other signal in order to count the instructions made. The decoding of operation 3 requires 5 signals from the ST control pipeline to activate the capture of data from the register, multiplexer and from the accumulator.

Finally operation 4 is shown in figure 4, this operation requires 9 pulses for the decoding of the operation. The control signal "xi1" and "deco 4" allow the capture of the operations in the register in order to make the multiplication.

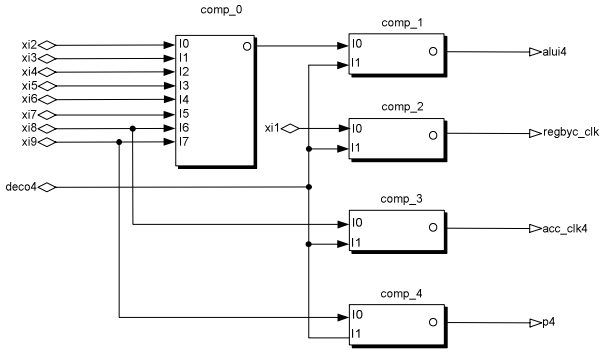


Figure 4. Arithmetical operation 4

4. ST pipeline control

The control units developed with ST circuits are frequently composed of micropipeline structures. These were proposals made by Ivan Sutherland at the end of the 1980s [1]. An ST pipeline structure consists of a successive series of control blocks that possess request and acknowledge signals, interconnected block by block, to facilitate the movement of information along all the circuit in a controlled and phased way. This type of structure is the fundamental base of the control circuits that are presented in this work.

4.1. Four phase pipeline structures

The majority of synchronous circuits have a data path by which the data is transferred during a process. A typical synchronous data path is formed by pipelines that have registers in their entrances and exits to store data that are processed by combinatorial circuits. These registers are controlled by clocks. On the other hand, asynchronous designs use two methods to control the transfer of data: bundled data [1] and dual rail [7].

Pipeline architectures operating in bundled data consist of delegating the control over the validity of the data in the signal, in such a way that it operates together with the acknowledge signal of the corresponding protocol.

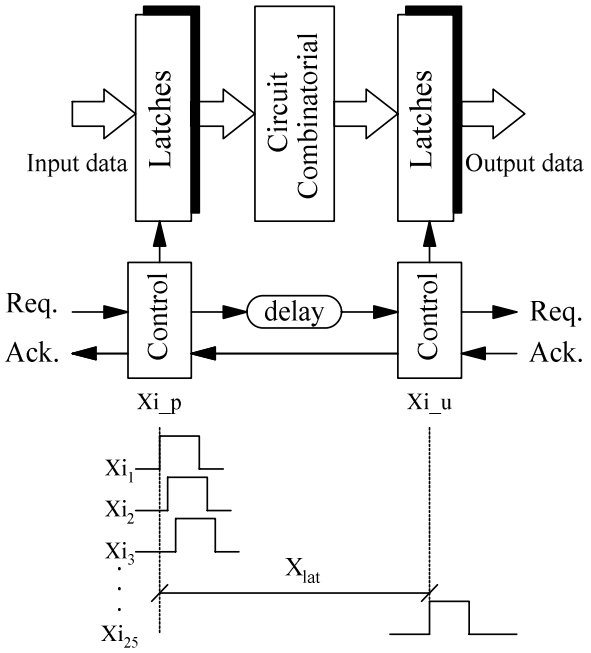


Figure 5. A bundled data pipeline

In figure 5 the illustration shows an architecture that follows the codification method in bundled data. One can see that it has a request signal "Req", an acknowledge signal "Ack" and a data bus. A combinatorial logic block sends a request signal to the next block when the data is available, and this then sends a recognition signal back to the previous block, to indicate that the data has been received and is available for the next transfer. With this method we can completely separate the interface part from the combinatorial. In this way we can interact separately with both.

For the synchronization of the pipeline circuit to operate correctly the delay between the exit request signal of each block and the entrance request of the next stage should have an equal value at the time of the combinatorial circuit process [8].

The advantage of this method is its simplicity. However, it has the inconvenience of operating over the maximum processing time for the combinatorial circuit. In this sense we fail to take advantage of quicker operating times.

4.2. Delay Macros

The implementation of delays in reconfigurable circuits is achieved through a macro in FPGA editor, as shown in figure 6.

The slice of the FPGA virtex II is composed of 2 LUTs and 2 latches. For the implementation of the delay the LUT <G> is used in the upper part of figure 6. It has a logic depth of 1, between the entrance "s_in_ibuf" and the output "s_sal_obuf"".

The delay total Δ_{TOT} is composed of two classes of intrinsic delays of the FPGA that are the logic delay and that introduced by the interconnection path or route, these, in turn subdivide into different partial delays described in the equation 1.

$$\Delta_{TOT} = \delta_{PI} + \delta_{LUT} + \delta_{PO} + \delta_{RUT} \qquad (1)$$

Where:

δ_{PI} is the propagation delay between the entrance and exit pad of the tiopi module, with a value of 0.825 ns.

δ_{LUT} is the combinational delay between the entrances of the LUTs F/G at the exits X/Y of the tilo module, with a value of 0.439 ns.

δ_{PO} is the propagation delay between the entrance and the exit pad of the tioop module, with a value of 6.107 ns.

δ_{RUT} is the propagation delay of the path or route of connection to the previous module.

Figure 6 shows the architecture of a delay of 9.637 ns with three logia levels corresponding to tiopi, tilo and tioop modules. The values correspond to the FPGA Virtex II Xilinx XC2V1000-4FG256 [9].

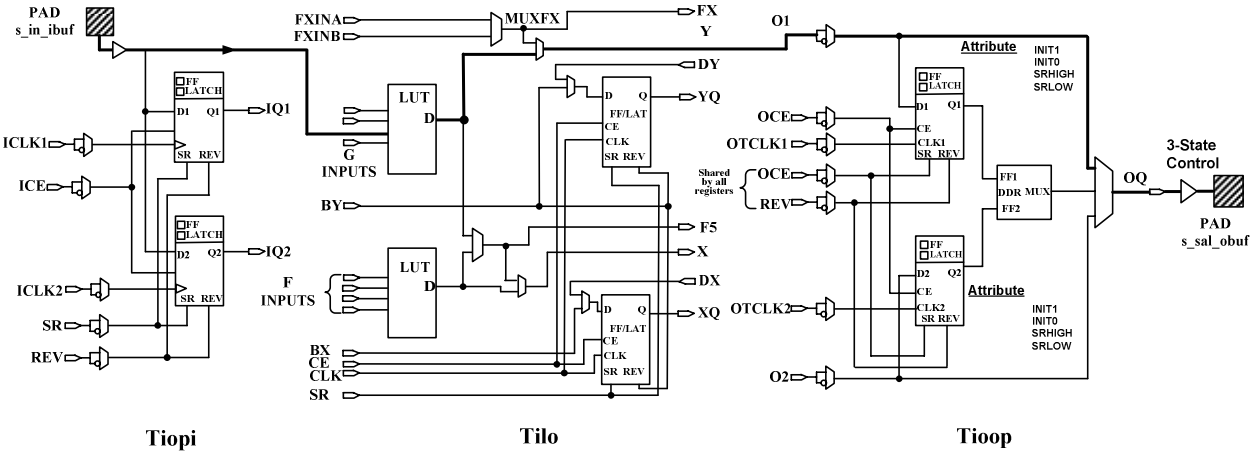


Figure 6. Implementation of delay macro in FPGA

To increase the delay various macros are connected in series, increasing the logic depth between the entrance and exit.

The figure 7 describes the characterization of 75 different delay modules, which shows that the total delay does not present a lineal behavior with respect to the number of macros of which it is composed, owing to the variation of the delay values of the FPGA routes. The table 2 presents 6 examples of the circuits characterized. One observes that the number of macros used in the implementation of the delay modules maintains a direct relationship with respect to the logic levels; this provides an approximation of the resources that will be used in the FPGA.

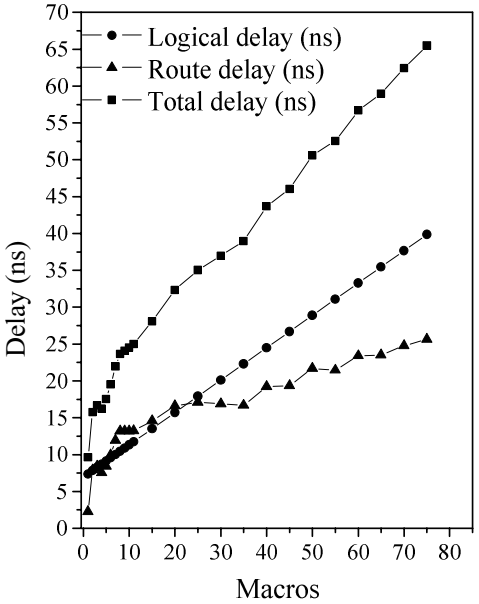


Figure 7. Total delays vs. macros

This same table shows 6 results of the measurements in the 75 delay modules on programming the FPGA. The value of T_{X1} indicates the entrance pulse (signal "s_in_ibuf"). T_{X2} is the exit pulse present in the circuit (signal "s_sal_obuf"). The value T_{real} represents the propagation delay present in each module (T_{X2}-T_{X1}) and the T_{ps} is the delay post-layout. The value of the delay in the FPGA tends to be less than the result measured during the simulation.

Although one should consider that the increase in the temperature in the FPGA alters the values of delay after the device has operated for a long period.

Macros	Logical level	T_{X1} (µs)	T_{X2} (µs)	T_{real} (ns)	T_{ps} (ns)
1	3	43.66	43.67	10	9.63
5	7	22.85	22.83	12	17.52
10	12	15.72	15.71	14	24.51
30	32	25.39	25.41	22	36.97
50	52	1.75	1.78	26	50.58
75	77	2.40	2.37	34	65.49

Table 2. Measurements of the delay circuit

4.3 Implementation of the ST control pipeline

A control element pipeline ST is used to regulate the data flow through a segmented system. This can also be used as an activation control for the different stages of the system. The tasks carried out by each one of the blocks are independent of one another and the time of the stage can be different. [10]:

In a pipeline structure (figure 5) of 25 asynchronous control blocks, the process of data transfer begins with the control pulse 'Xi_1' and finalizes with 'Xi_{25}',in such a way that the processing time X_{Lat} is related to the difference between the last (Xi_u) and the first Xi (Xi_p) as is seen in the figure 5. For the characterization of these structures and to anticipate the number of macros (Nω) to use in a ST pipeline of a specific size, we use the equation 2.

$$N\omega = N\tau * (ret\_2 + ret\_1) \quad (2)$$

Where:

Nt is the number of control blocks within the structure minus 1.

Ret_1 is equal to the delay in feedback, with a value of 1.

Ret_2 establishes the calculation time between the blocks, for this characterization, with a value of 3.

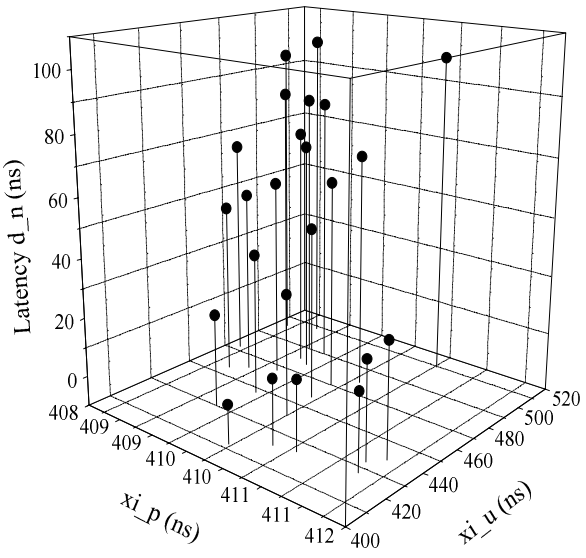


Figure 8. Latency of different pipelines

Figure 8 shows a diagram with the characterization of the latency for 25 pipeline control structures of different sizes. In that of the group of xi_p it is the time form the start of the first control pulse generated by the first, and xi_u is the start of the final pulse produced by the last control. D_n is the time that passes in order to generate all the control pulses, or what it takes for the complete process to send data or latency.

The value of the rising edge for the first pulse is found to be around 409.71 ns and the rising edge of the final pulse is found to be in the range of 412.75 to 509.57ns. The time cycle for each one of the structures demonstrates a gradual behavior on increasing the number of elements of which it is composed, for this motive a pipeline control of 2 elements implemented in the VIRTEX II is of 3.03ns and for 25 elements 99.30 ns.

5. Execution of ST ALU

In synchronous circuits the measurement of operations is traditionally based on operation frequency of the clock. In the case of ST circuits the speed depends on the delays incorporated into the ST control to modulate the transfer and activation of the circuit operations.

Each type of operation was characterized varying the delay and the number of millions of instructions per second (MIPS) executed was observed. It displayed a diminution when increasing the delays. As shown in figure 9.

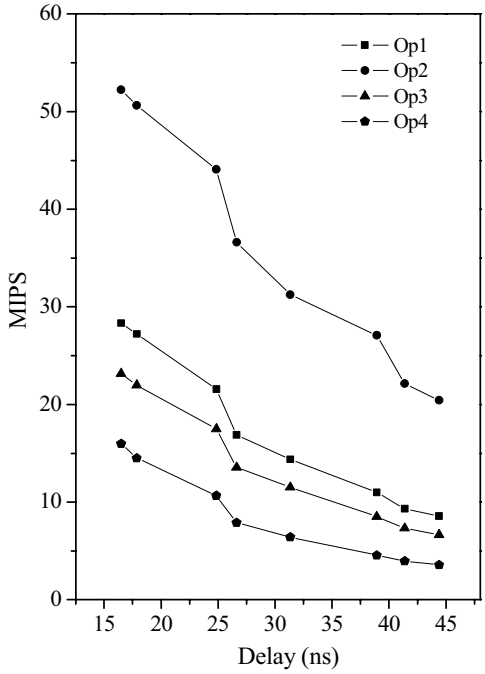


Figure 9. MIPS vs. delay (ns)

The latency by type of operation which was displayed by the ALU when varying the number of delay macros was more significant for the operations that require a greater number of pulses of control Xi during the process (type 3 and 4). The behavior of the latency against the number of macros by type of operation is shown in figure 10.

In synchronous circuits the fan-out of the global lines tends to be greater, especially the clock lines, enable, reset etc. In asynchronous circuits the

interconnection lines are local and therefore the level of fan-out is smaller. For ST ALU 95% of the lines have a fan-out smaller than 20 and net delays of less than 4ns, as shown in figure 11.

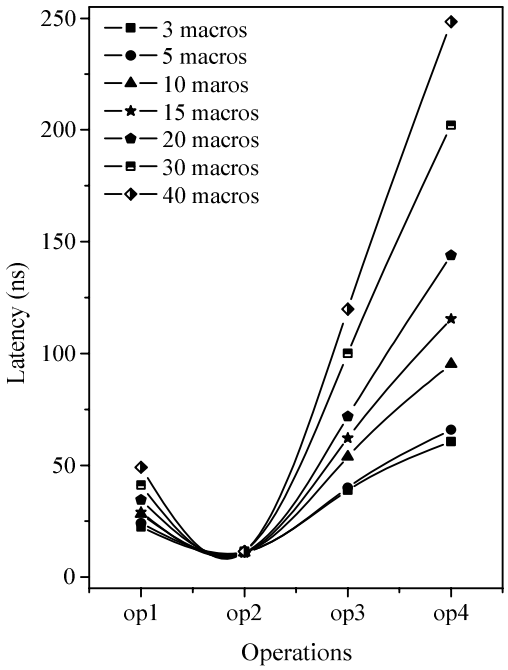


Figure 10. Latency of the ALU operations

The line of reset that it is not drawn has a fan-out of 92 and a delay of 6 ns.

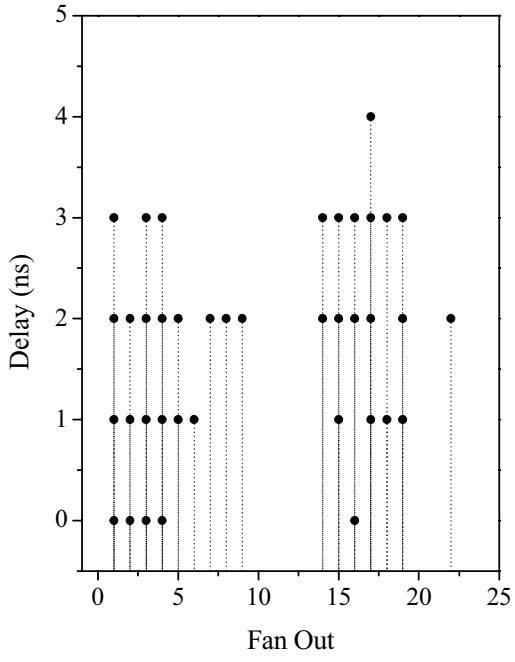


Figure 11. Fan-out vs. net delay

The results of the implementation of the circuits in the FPGA virtexII are summarized in the figure 12. Where the ALU occupy more LUTs and registers.

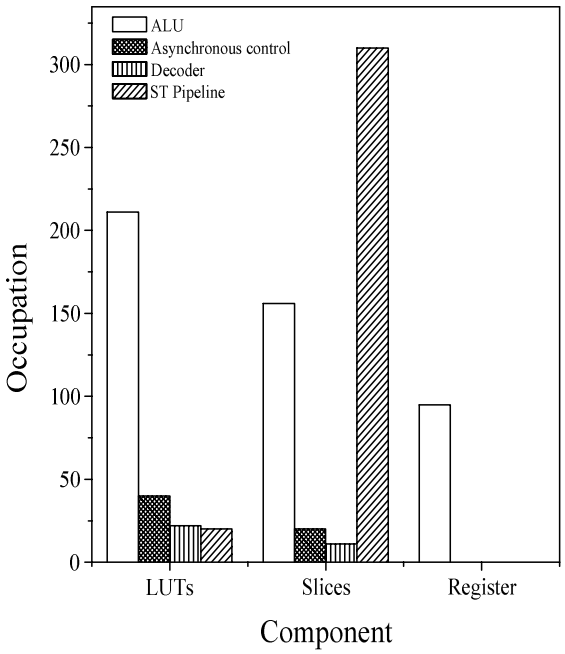


Figure 12. Occupation of the ST ALU

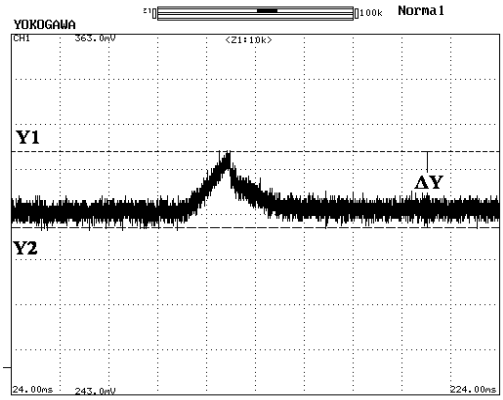
6. Instantaneous Current

In order to measure the instantaneous current a current probe was used. This is an indirect method of measurement that detects the electromagnetic variation of the feed cable. The results of the measurement of the instantaneous current can be seen in figure 13.

The measurements were made on an Avnet evaluation card. An ammeter was connected in series with the main feeder cable and the current probe monitored the same cable. The measurements of instantaneous current are registered on an oscilloscope.

The results obtained of the measurement of current registered by the ammeter for ST ALU were 494 mA. With respect to its synchronous counterpart the current measured with the ammeter was 496 mA in execution. The measurements with the current probe, in the ST ALU detected a change of voltage peak to peak of 25 mV, as is observed in figure 13. In the case of its synchronous counterpart a voltage increase was observed, peak to peak, of 24 mV, throughout the time that the circuit stayed operating.

As was observed in the synchronous ALU the changes of instantaneous current were more than those of the ST, from which we can deduce that the consumption of a synchronous circuit is greater.



Y1 323.850 mV	Y2 323.850 mV	ΔY 25.2 mV
CHI 10:1	15.0 mV/div	DC Full

Figure 13. Instantaneous current of the ST ALU

7. Conclusions

This article describes to the implementation of a Self-Timed ALU in reconfigurable circuits. In addition, it suggests some ideas for the design of these circuits in FPGAs, as well as the characterization of the measurements of the delays generated in real time and the occupation of the resources in Virtex II.

The ALU has the characteristic to activate with an external pulse and eliminates the dependency on a global clock.

It presented an analysis of the effect of the delay macros with different values on the behavior of the ST ALU with respect to the number of operations executed per second. A small reduction was observed in the consumption in the power supply line during the execution of an operation of the ST ALU compared to its synchronous counterpart. The feasibility was tested of making a fast prototype of ST circuits in a synchronous tool.

8. Acknowledgment

This work has been financed by the National Advice of Science and Technology of México (CONACYT).

9. References

[1] I. E. Sutherland, "Micropipelines", *Communications of the ACM*, vol. 32, No. 6, June 1989, pp. 720-738.

[2] P. Kudva and V. Akella, "Testing two-phase transition signalling based self-timed circuits in a synthesis environment," in *Proceedings of the 7th International Symposium on High-Level Synthesis*, IEEE Computer Society Press, May 1994, pp. 104–111.

[3] S. B. Furber and P. Day, "Four-phase micropipeline latch control circuits," *IEEE Transactions on VLSI Systems*, vol. 4 June 1996, pp. 247–253.

[4] A. J. McAuley, "Four State Asynchronous Architectures", *IEEE transactions on computers*, vol. 41, No. 2, Feb. 1992.

[5] S. B. Furber and J. Liu, "Dynamic logic in four-phase micropipelines", in *Proc. Interna-tionalSymposium on Advanced Research in Asynchronous Circuits and Systems*, IEEE Computer Society Press, Mar. 1996.

[6] Kees van Berkel and Arjan Bink. "Single-track handshaking signaling with application to micropipelines and handshake circuits", In *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems,* IEEE Computer Society Press, March 1996, pp. 122-133.

[7] M. Dean, T. Williams, and D. Dill, "Efficient self-timing with level-encoded 2-phase dual-rail (LEDR)," in *Advanced Research in VLSI* (C. H. S´equin, ed.), MIT Press, 1991, pp. 55–70.

[8] R. Kelly "Asynchronous Design Aspects of High-Performance Logic", *Thesis*. University of Manchester. Department of Computer Science UK, 1995.

[9] Co. Xilinx "Virtex-II Platform FPGA" *User Guide*, 1-800-255-7778 UG002 (v1.5). www.xilinx.com. Dec. 2002.

[10] D. A. Gilbert and J. D. Garside, "A result forwarding mechanism for asynchronous pipelined systems," in Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems, IEEE Computer Society Press, Apr. 1997, pp. 2–11.