Comparing the Performance of 4-bits Adders Topologies on Sub 32nm Bulk CMOS Technologies

Albano Borba, Vagner Rosa
Centro de Cilncias Computacionais -C3
Universidade Federal do Rio Grande - FURG
Email: albano.b06@gmail.com, vsrosa@gmail.com

Cristina Meinhardt
Departamento de Informtica e Estatstica
Universidade Federal de Santa Catarina
Email: cristina.meinhardt@ufsc.br

***Abstract*—This work presents a comparative study of some adder models analyzing characteristics such as delay and power at electrical level for predictive CMOS technology of 16 and 32nm high performance. Four 4-bit architectures are compared: Ripple-Carry, Carry-Lookahead, Carry-Select and the Kogge-Stone Tree adder. For 32nm, the Carry-Select and Kogge-Stone Tree adders showed the best results for maximum delays, around 40% faster than Ripple-Carry. However, the architectures presented more than twice the average energy consumption. Considering power and performance, the Carry-Lookahead architecture presents the best cost-benefit ratio.**

I. Introduction

THE integrated circuits (ICs) have revolutionized the world of electronics and are present in the most varied electronic equipment used today. They are composed of several semiconductor devices on a silicon wafer. Due to the miniaturization of the components, we can obtain quite complex electronic architectures in a small area. With the components in much smaller scales, in the nanometric order, the current electronic products present a great performance and an increase in the number of functionalities.

Adder circuits have a key role in the operation of any electronic system, from the simplest controllers to the most complex microprocessors. These circuits are the focus of several researchers, because they are part of the critical path of most computer systems [1] [2].

In addition, every day the capture and processing of digital images and videos are present in our daily lives, from medical, industrial and even entertainment applications. Images in 10Mpixel (photo) and video in Full HD resolution, which are typical image and video capture formats currently, can generate 30MB and 300MB/s respectively. Therefore, dealing with this massive amount of data requires that images and videos are capable of processing high taxes. Image and video compression consists of a set of video and image processing algorithms standard in most devices to reduce storage or transmission required for entertainment applications. For industrial applications, the processing of videos in the captured rates by sensors require the use of dedicated hardware with high processing. Graphic processing involves a large amount of arithmetic processing, exploring in deep the exploiting to the maximum the sum and multiplication modules. In this context, optimized design of adder circuits can contribute to the overall system performance increase.

There are different architectures that implement n-bit addition circuits. Each approach has well-exploited advantages and disadvantages in relation to area, delay and power. The main project in which this work is inserted explores some architectures, seeking the development of dedicated sum modules for image processing. This project is divided into two phases, an initial, exploratory, to define the performance of binary adders, which are building blocks more fundamental for audio and video processing. This is critical to the development of the second phase, which is image processing itself.

Thus, composing the first phase of this project, the objective of this work is to explore the project of n-bits adders using predictive models of MOSFET transistors in nanotechnologies for the design of CMOS integrated circuits. The goal is to achieve the best performance or lower energy consumption per operation in order to obtain processing rates compatible with those required for images and videos.

II. Methodology

Matrix calculations, mainly through the application of filters, are key points of image processing, making the study of adders in digital circuits the first step of this study. This work was chosen to use in a first approach the predictive CMOS technology of 16nm and 32nm high performance [3][4]. The analysis of the electrical characteristics considers the delays and the energy consumed in each transition. This analysis is made considering the SPICE descriptions of the architectures and performing electrical simulations with the NGSPICE tool [5]. All designed architectures have been logically validated using auxiliary scripts through the Python language. To allow the comparison between the architectures this work adopts the execution flow shown in Fig. 1. These steps will be detailed in the next subsections.

All the evaluated architectures adopt as internal module the mirror CMOS full adder circuit, shown in Fig. 2. This architecture is called CMOS, by exploring the two logically and topologically complementary planes [2].

The full adder accepts the ”A”, ”B” entries, which refer to the operands, and ”Cin”, which represents the carry signal from the previous bit of the sum. The circuit generates the signals ”Sum”, with the result of the sum, and ”Cout”, for the generated carry-out. The latter are represented by equations:

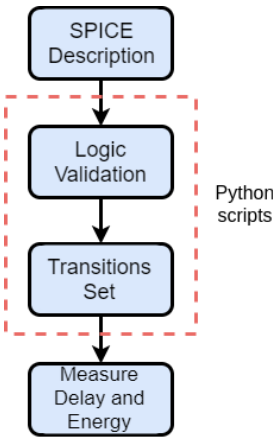

Fig. 1. Execution flow

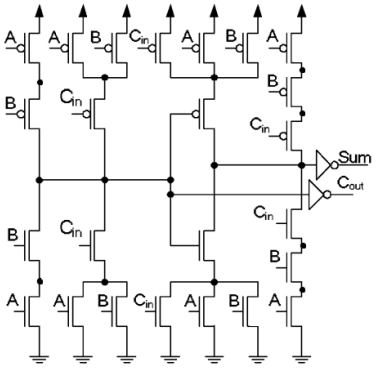

Fig. 2. FA - CMOS

$$Sum = A \oplus B \oplus Cin \tag{1}$$

$$Cout = A.B + A.Cin + B.Cin \tag{2}$$

The choice of this circuit for the complete sum of 1 bit is due to the fact that it is the most frequently available in the commercial cell libraries and because it is adopted in the logical and arithmetic units of processors, having good performance compared to other architectures. This 1-bit module is used for the construction of n-bit adders. As focus of this work, the selected four different architectures for the implementation of n-bit adders: Ripple-Carry adder, Carry-Lookahead adder, Carry-Select adder and the Kogge-Stone Tree adder. The main details of each of these architectures are presented in the next sub-sections.

1) Ripple-Carry Adder: it is the simplest one that can be built. As shown in the example of a 4-bit Ripple-Carry adder of Fig. 3, for an n-bit circuit this adder demands n FA modules connected in series. The main disadvantage of this architecture is the critical path. In this case, the longest path for a signal to be generated by the circuit depends on all modules processing their carry propagation values so that the value of the most significant bit of the sum can be generated.

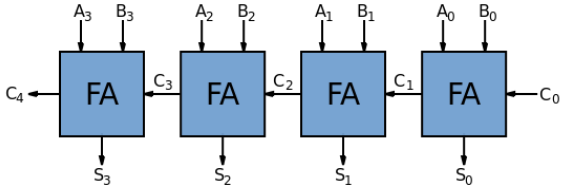

Fig. 3. Ripple-Carry Adder - 4 bits

2) Carry-Lookahead Adder: The next adders to be addressed, adopt two signals as an aid in their project. These signals are used to accelerate the propagation of carry to the next bits, offering alternatives to reduce the critical path problem of Ripple-Carry. Basically, there are two main signs: generate and propagate.

The first one is Generate (G), which will be '1' when carry-out is '1' independent of carry-in, as shown in equation (3). The signal Propagate (P) will be '1' when the inputs are different, according to equation (4), representing that the carry-out will be equal to the carry-in. These signals can still be related to equations (1) and (2) of the FA so as to obtain the Sum and Cout signals as a function of P and G, resulting in:

$$G = A.B \tag{3}$$

$$P = A \oplus B \tag{4}$$

$$Sum = A \oplus B \oplus Cin = P \oplus Cin \tag{5}$$

$$Cout = A.B + A.Cin + B.Cin = G + P.Cin \tag{6}$$

Differently the Ripple-Carry model, where each FA expects the carry-out of the previous one to make its sum, in the Carry-Lookahead adder each module computes its own carry-in bit independently. The Fig. 4 shows the organization of this model to 4 bits, where no FA communicates directly. In general, an n-bit Carry-Lookahead adder requires n adders and the generation of the carry propagation signal.

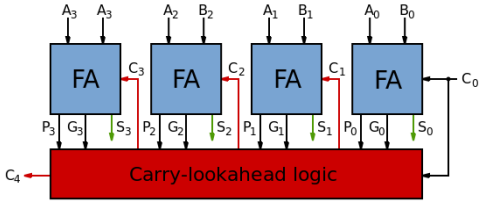

Fig. 4. Carry-Lookahead Adder - 4 bits

Using the auxiliary signals, the Carry-Lookahead adder can obtain the carry-out signals for each bit through equations (7), (8), (9) and (10), defined as:

$$Cout_0 = G_0 + P_0.Cin_0 \tag{7}$$

$$Cout_1 = G_1 + P_1.G_0 + P_1.P_0.Cin0 \tag{8}$$

$$Cout_2 = G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.Cin0 \tag{9}$$

$$Cout_3 = G_3 + P_3.G_2 + P_3.P_2.G_1 + P_3.P_2.P_1.G_0 + P_3.P_2.P_1.P_0.Cin0 \tag{10}$$

3) Carry-Select Adder: The Carry-Select adder consists basically of two Ripple-Carry adders and a multiplexer. As shown in the Fig. 5, for the sum of 4 bits two parallel adders are needed, where the first is fed with the carry-in '0' and second with carry-in '1'. Then the multiplexer selects the correct signal from the initial carry-in of the sum.

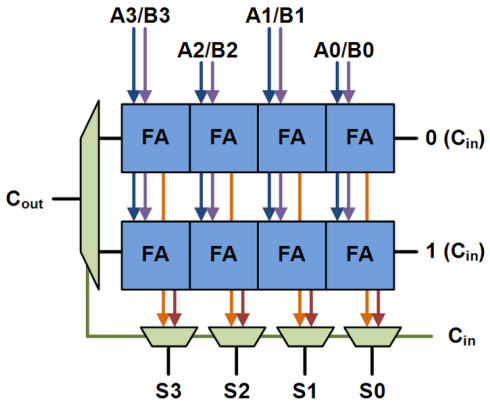

Fig. 5. Carry-Select Adder - 4 bits

4) Kogge-Stone Tree Adder: Tree-based adders architectures are proposed for multi-bit adders, that is, adders larger than 4 bits. Among the alternatives proposed in the literature, the Kogge-Stone adder stands out for its performance [6]. The Fig. 6 shows a 16-bit Kogge-Stone tree based adder. Each cell type in this architecture is responsible for a different equation and are divided by layers, with the number of layers equal to $log_2 n$, where n represents the number of bits in the adder.

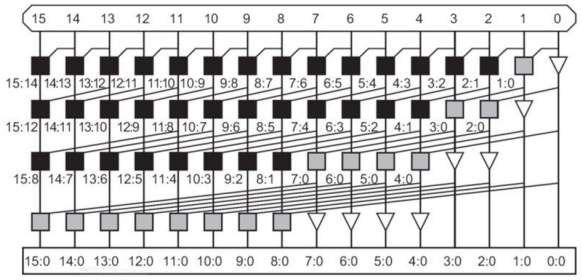

Fig. 6. Kogge-Stone Tree Adder - 16 bits

Initially, all the propagate and generate signals for each bit are calculated. The other equations are represented by:

$$Sum_i = P_i \oplus Cin_i \tag{11}$$

$$G_{i:j} = G_{i:k} + P_{i:k}.G_{k-1:j} \tag{12}$$

$$G_{i:j} = G_{i:k} + P_{i:k}.G_{k-1:j} e P_{i:j} = P_{i:k}.P_{k-1:j} \tag{13}$$

A. Performance Evaluation of Transition Arcs

With the adders described and logically validated, the third part of the execution flow starts, where are simulated the cited models for 4 bit, since the simulation and study of its variables demand an analysis on the performance of the transition arcs of the circuit. These arcs consist of pairs of inputs (each input being represented by two binary numbers with n bits) that differ in only 1 bit but which generate distinct outputs (the sum of the two numbers), these being them the n bits of *sum* and last *carry-out*.

Energy consumed and delay are extremely important parameters for evaluating electrical circuits and developing improvements. To verify the efficiency using these variables, from the set of transition arcs, another script mounts SPICE files for the simulation of the propagation times and the energy consumption for the evaluation of the performance of the transition arcs. The SPICE file is divided into three parts, a first one where the sources that make up the circuit are declared. Each input bit is described as a DC source, with the exception of the variable bit that is simulated in low-high and high-low states. In the second part, the adder is described, according to the model to be used in the evaluation, generating the output signals. Finally, the delay times of the signals and their energy consumption to be compared are calculated. The scripts are written on Pyhton and automatically find all arcs and also generate the description files to measurement of power and delay for n-bit adders. In this case, for 4-bits adders, the total arcs found and evaluated are 1024 combinations.

III. Results

This work analyzes the longer times of delay and the respective energy consumption for the adder models Ripple-Carry, Carry-Lookahead, Carry-Select and Koge-Stone Tree with 4 bits. The product evaluation between power and delays (the Power-Delay-Product, PDP) is also presented. The results for 32nm are shown in Tables I and III, for 16nm are shown in II and IV. The normalized values for PDP are presented in Fig. 7.

Tables I and II show the five largest delay times and the mean signal propagation delay for each of the four architectures, in nanoseconds (ns), in 32nm and 16nm technologies respectively considering all the possible transistion arch on 4-bit adders. For 32nm, it is observed that the Kogge-Stone Tree and Carry-Select adders have the lowest maximum delay times among the 4 models, but Kogge-Stone Tree has the longest average delay time. This is due to the complexity of the circuit, which is often not compensated in performance for adders with less than 16 bits. It is also noted that the Ripple-Carry adder showed the shortest mean delay time in the simulations, but it presents the worst cases due to the absence of acceleration of the carry propagation. Comparing the worst delays with the Ripple-Carry adder, the Carry-Lookahead architecture has a 35% reduction, with Kogge-Stone Tree and Carry-Select around 40% faster. Similar behavior was observed for 16nm technology, however, the architectures Kogge-Stone Three and Carry-Lookahead are presented the lowest delays and worst average.

Similarly, Table II shows the five largest simulated energy intakes for each model and technology in femtojoules (fJ). The Carry-Lookahead adder showed the lowest consumptions, but for 16nm the Carry-Select adder shows the lowest average

TABLE I
COMPARISON OF SIGNAL PROPAGATION TIMES FOR 32NM

Model	Longer delay times (ns)					Avg. (ns)
Ripple-Carry	0.932	0.922	0.922	0.921	0.918	0.297
Carry-Lookahead	0.613	0.613	0.613	0.613	0.613	0.387
Carry-Select	0.543	0.543	0.543	0.543	0.543	0.360
Kogge-Stone Tree	0.549	0.549	0.549	0.549	0.541	0.469

TABLE II
COMPARISON OF SIGNAL PROPAGATION TIMES FOR 16NM

Model	Longer delay times (ns)					Avg. (ns)
Ripple-Carry	0.806	0.806	0.790	0.790	0.786	0.249
Carry-Lookahead	3.565	3.565	3.565	3.565	3.565	0.637
Carry-Select	0.404	0.404	0.404	0.404	0.404	0.337
Kogge-Stone Tree	1.393	1.393	1.393	1.393	1.357	0.497

value. The Kogge-Stone Tree achieved more than double the average energy consumption compared to other models. due to its greater use of hardware, i.e. elevated number of gates compared to the others one.

TABLE III
COMPARISON OF ENERGY CONSUMPTION FOR 32NM

Model	Higher energy costs (fJ)					Avg. (fJ)
Ripple-Carry	5.588	5.588	5.588	5.588	5.588	1.90
Carry-Lookahead	3.888	3.888	3.888	3.888	3.887	1.61
Carry-Select	6.916	6.916	6.916	6.916	6.916	4.14
Kogge-Stone Tree	7.515	7.515	7.515	7.515	7.508	4.15

TABLE IV
COMPARISON OF ENERGY CONSUMPTION FOR 16NM

Model	Higher energy costs (fJ)					Avg. (fJ)
Ripple-Carry	3.587	5.587	3.586	3.586	3.585	1.81
Carry-Lookahead	1.733	1.733	1.732	1.732	1.732	0.76
Carry-Select	0.353	0.353	0.353	0.353	0.353	0.15
Kogge-Stone Tree	9.107	9.107	9.106	9.106	9.037	5.19

Power and delay are extremely important parameters for evaluating electrical circuits and developing improvements. To verify the efficiency using these two questions, this work also evaluates the Power Delay Product (PDP). The Fig. 7 shows the graph with the normalized values of PDP, between 16nm and 32 nm approaches, considering all transition arcs and also the average value, where the average power and the average delay of each architecture in attojoules (aJ) are adopted. PDP results shows that, for 32nm, Carry-Lookahead adder presents the better tradeoff between power and delay, showing a good improvement in relation to the other adders in worst case. Considering the average PDP, Carry-Lookahead and Ripple-Carry presented similar behavior. For 16nm, Carry-Select showed a significant improvement, both in the average and worst case, for PDP results, since the Kogge-Stone adder showed the highest values of PDP in this same approach proving its low efficiency for few bits. The graph in Fig. 7 with the normalized values presents the relations between the commented data.

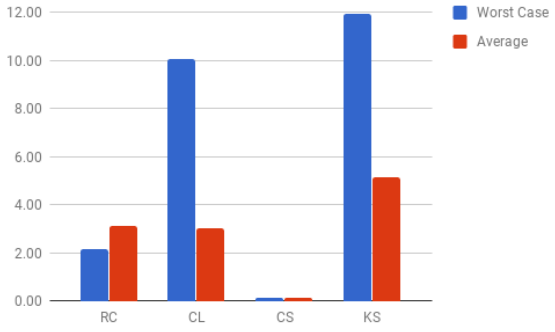

Fig. 7. Normalized PDP comparison

IV. CONCLUSIONS

In this work, adders circuits were presented and simulated in 16nm and 32nm high perfomance technologies, from 4 models: Ripple-Carry, Carry-Lookahead, Carry-Select and Kogge-Stone Tree. A framework was developed to aid in the exhaustive evaluation of transition arcs, measurement of delays, and energy consumption by adders. The models presented expected characteristics regarding the observed variables. The Ripple-Carry adder presented a low power consumption coupled with the lack of robustness of the model, presenting the greatest delays. The Carry-Lookahead model was more efficient than the first in terms of signal propagation time and the lowest average energy comsuption for 32nm. However, the Carry-Select adder presented the best values for 16nm.Finally, the Kogge-Stone Tree adder had a good delay time but a high power consumption for their 4-bit simulation. These architecture of n-bit adders probably reaches better results for 16 bits adders or more. Thus, the next steps are to perform the simulations for 8, 16, 32 and 64 bits with the same analysis of propagation delay and power consumption. Also, it is expected to describe the models of adders from other technologies besides the 32nm and 16nm bulk CMOS.

ACKNOWLEDGMENT

The authors would like to thank to CNPQ, CAPES and FAPERGS for supporting this project.

REFERENCES

[1] M. Alioto and G. Palumbo, Analysis and comparison on full adderblock in submicron technology,IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 10, no. 6, pp. 806823, 2002.
[2] C.-H. Chang, J. Gu, and M. Zhang, A review of 0.18um full adder performances for tree structured arithmetic circuits, IEEE Transactions on very large scale integration (VLSI) systems, vol. 13, no. 6, pp. 686695, 2005.
[3] W. Zhao and Y. Cao, New generation of predictive technology modelfor sub-45 nm early design exploration,IEEE Transactions on ElectronDevices, vol. 53, no. 11, pp. 28162823, 2006.
[4] PTM. Predictive Technology Model. http://ptm.asu.edu.
[5] P. Nenzi and H. Vogt, Ngspice users manual version 23, 2011.
[6] P. M. Kogge and H. S. Stone, A parallel algorithm for the efficient solution of a general class of recurrence equations,IEEE transactionson computers, vol. 100, no. 8, pp. 786793, 1973.