Power Reduction Techniques for Asynchronous Circuits

Kamran Saleh
k.saleh@ce.aut.ac.ir

Hossein Pedram
pedram@ce.aut.ac.ir

Mohsen Naderi
naderi@ce.aut.ac.ir

Department of Computer Engineering and Information Technology
Amirkabir University of Technology

Abstract: *Power reduction is one of the main reasons for designing asynchronous circuits. Asynchronous circuits are inherently capable of having low power consumption. In this paper, we introduce five methods to achieve even lower power consumption. We have applied these methods to some sample circuits and the experimental results show that power reduction of 20% to 41% is obtainable.*

Keywords: Asynchronous, Power Reduction

1- Introduction

The power consumed in CMOS digital circuits, in the order of importance, consists of three main parts:

- Dynamic power
- Short circuit power
- Static power

Approximately 99% of power is consumed as dynamic and short circuit currents, so neglecting the static power we can say that the power consumption of a circuit with no activity is almost zero.

In asynchronous circuits, there is no activity in those parts of a system that do not perform any logical function, so only active parts of a circuit consume power. On the other hand, in synchronous systems all registers receive the clock signal and consume power all the time.

To reduce the power consumption of asynchronous circuits we must reduce the activity of the circuit. For this purpose, we should determine the role of each section of the circuit in consuming power.

Section 2 describes the overall structure of asynchronous circuits and predefined templates. Section 3 shows our power estimation method and section 4 identifies the proportional role of each section of a circuit in total power consumption by examining two sample circuits. Section 5 describes power reduction methods and their impact on power and circuit size of sample circuits.

2- Structure of asynchronous circuits

Among the various methods of asynchronous design [1, 2, 3], the method introduced by Caltech Research Group [4] is capable of having better performance, lower power consumption and being more adaptive against environment variations. This method is based on QDI (Quasi Delay Insensitive) timing model.

Due to difficulties in this method and the lack of automatic CAD tools there is an increasing trend to use predefined templates [5, 6, 7].

At present, most QDI circuits are designed using PCHB (Pre-Charge logic Half-Buffer) and PCFB (Pre-Charge logic Full-Buffer) templates [8]. Figure 1 shows the overall structure of these templates. The template performs 4-phase handshakes on both input and output channels. The data in these channels are encoded in dual-rail form [9].

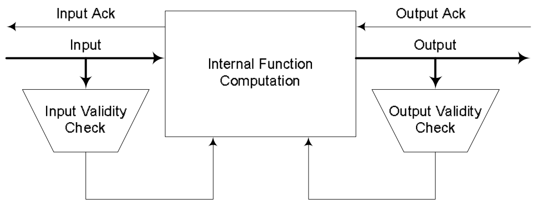

Figure 1: Structure of PCHB/PCFB templates

3- Power estimation method

There are various methods for power estimation and one of them is Transition Counting [10, 11]. Using this method, we do not need to use analog simulators such as Spice and the simulation time will decrease dramatically. In circuits with monotonous and regular fan-out like data-paths, by counting the transitions on the gate outputs, we can estimate power consumption with an accuracy of more than 90%. But in case of control logic, which has a random nature the accuracy falls below 60% [10], because in this method the fan-out does not contribute to transition counting. To solve this problem we can count the transition on the input of gates [11]. Using this method the counter will count three times for one transition that occurs on the output of a gate connected to three gates and once for a transition on the output of a gate connected to one gate (Figure 2).

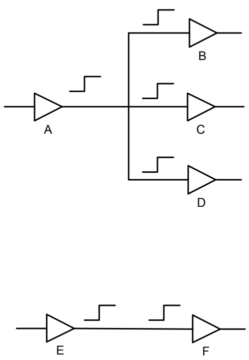

Figure 2: Counting the transitions on inputs

We synthesized our sample circuits down to the switch level and we counted the transitions on the gates of the transistors to gain high accuracy.

4- Power consumption of each portion

Table 1 shows the number of transitions and the transistor count of three main parts of a 4-bit PCFB buffer. The transitions are counted by applying random inputs.

Table 1: Number of transistors and transitions in three main parts of a 4-bit PCFB buffer.

4-Bit Buffer	Total	Input Validity Check	Output Validity Check	Internal Circuit
Transistors	184	36 19.5%	38 20.6%	110 59.7%
Transitions	131	28 **21.3%**	30 **22.9%**	73 **55.7%**

Table 1 shows that almost 44% of total transitions have occurred in I/O-validity-check circuits. This value increases proportionally to the number of input and output lines.

As another example, we depicted a 4-bit Galois Field Adder implemented in PCHB with two 4-bit inputs and a 4-bit output. Table 2 shows the result where the role of IO-validity parts increases to 55% of total transitions.

Table 2: Number of transistors and transitions in three main portions of a 4-bit PCHB Galois Field Adder for random inputs.

4-Bit GF Adder	Total	Input Validity Check	Output Validity Check	Internal Circuit
Transistors	228	72 31.5%	38 16.6%	118 51.7%
Transitions	156	56 **35.8%**	30 **19.2%**	70 **44.8%**

Comparing the results of Tables 1 and 2, we can conclude that the power consumption of input-validity circuit in GF Adder is twice as that of similar portion in 4-bit Buffer. This is due to doubling the number of input bits of GF adder.

5- Power reduction methods

So far, we explained the power consumption in PCHB/PCFB based circuits and the approximate role of each portion in total power consumption. We cannot exploit the internal computation part of a circuit to reduce power, but we can make some changes in validity-check circuits to reduce the number of transitions.

5-1- Elimination of input-validity-check

In a system consisting of PCHB/PCFBs, there are two validity-check (completion detection) circuits on the both ends of each communication channel (Figure 3). Therefore, the simplest way to reduce the power consumption and circuit size is to eliminate one of them.

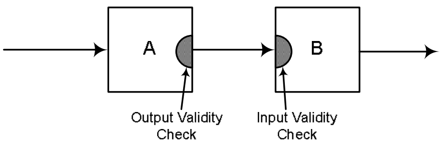

Figure 3: Validity-check circuits

We can eliminate the input-validity-check from circuit B and use the result of validity-check in circuit A [12].

As an example, consider a FIFO with 15 places of 4-bit words. The simplest way to design such a circuit is to connect 15 PCFB 4-bit buffers in a chain as shown in Figure 4.

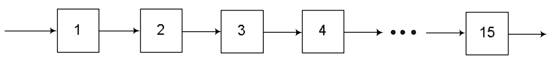

Figure 4: Simple 15 word FIFO

New data words enter from buffer 1 and traverse all buffers to reach the last buffer and exit the FIFO. We applied 15 random inputs to this circuit and waited for all of them to exit the FIFO. Table 3 shows the results of this experiment before and after eliminating the input-validity-check circuits of all buffers except the first one. FIFO-1 is the circuit with full validity-check circuits in inputs and outputs and FIFO-2 is the circuit after eliminating the input-validity-check circuits.

Table 3: Experimental results of FIFO before and after eliminating the input-validity-check circuits

FIFO	FIFO 1	FIFO 2
Transistors	2760	2256 (**-18.2%**)
Transitions	30974	23385 (**-24.5%**)

According to Table 3, we have reduced the number of transitions (power consumption) by 24.5% and the number of transistors by 18.2%.

5-2- Elimination of unnecessary handshakes

In the circuit of Figure 4, each data word passes all buffers on its way from input to output of the FIFO and causes more than 15 handshakes to occur. We can omit the redundant handshakes in order to reduce the activity of the circuit. One way is to design the circuit in such a way that a controller places each data word in its defined room and then another controller picks it up (FIFO 3 in Figure 5).

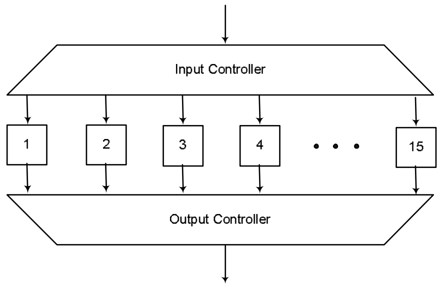

Figure 5: The new design of FIFO to reduce the number of handshakes (FIFO 3)

This way, each data words will pass three PCFB elements and the total number of transitions should reduce to 1/5 of its original value, but the experimental result does not indicate this. As it is seen in Table 4, we have reduced the number of transitions by 20% and the circuit is 2.73 times larger than the original FIFO. This is due to large fan-out of the input controller circuit and large fan-in of the output controller circuit.

Table 4: Experimental results of FIFO 3

FIFO 3	Input Controller	Buffers	Output Controller	Total
Transistors	2943	2760	1833	7536 (**+173%**)
Transitions	12876	3480	8449	24805 (**-20%**)

5-3- Special circuits vs. pre-defined templates

The input controller of Figure 5 consumes more than half of the total power in FIFO 3, so we can concentrate on this part to reduce the power consumption. We can omit this part totally and replace it with a shared bus (Figure 6).

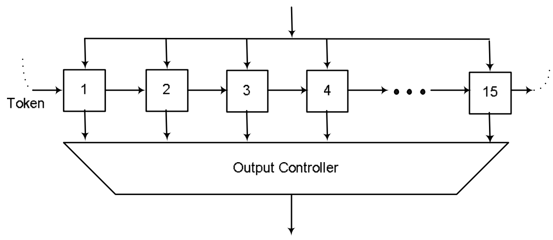

Figure 6: Shared bus instead of Input controller (FIFO 4)

The buffers of figure 6 have two other ports to exchange a unique token. Initially the left-most buffer has the token and after reading the first word, it passes the token to its right neighbor. Each time, only one buffer gets the data from the shared bus and acknowledges. After the assertion of the acknowledgement signal, the bus goes to neutral state and only after this time, the token can be passed to another buffer. The timing of this buffer does not match the PCHB/PCFB templates [5, 6, 7], so we must synthesize it by using Caltech's method [4].

Table 5: Experimental results of FIFO 4

FIFO 4	Buffers	Output Controller	Total
Transistors	3374	1833	5207 (**+88%**)
Transitions	9908	8449	18357 (**-40.7%**)

Regardless of its difficulties in the synthesis stage, designing special circuits has a great impact on power consumption of this circuit. As Table 5 shows, the power consumption of the circuit shown in Figure 6, named FIFO 4, is 41% less than that of the first version of FIFO.

5-4- Integrating back-to-back circuits

We can reduce the handshakes in data-paths that consist of more than one functional element by combining the adjacent modules.

As an example, consider the 4-bit Galois Field multiplier [13] described in the following equations:

$$c_0 = a_0b_0 + a_3b_1 + a_2b_2 + a_1b_3$$
$$c_1 = a_1b_0 + (a_0 + a_3)b_1 + (a_2 + a_3)b_2 + (a_1 + a_2)b_3$$
$$c_2 = a_2b_0 + a_1b_1 + (a_0 + a_3)b_2 + (a_2 + a_3)b_3$$
$$c_3 = a_3b_0 + a_2b_1 + a_1b_2 + (a_0 + a_3)b_3$$

Due to the restriction on number of transistors that may be connected in series, this multiplier cannot be implemented in the CMOS network of the computation part of a single PCHB/PCFB template. Therefore, we implemented this function as two back-to-back PCHB modules and named it *Mult 1* (Figure 7).

The left module computes the products of the polynomials and the right module computes the sum of products.

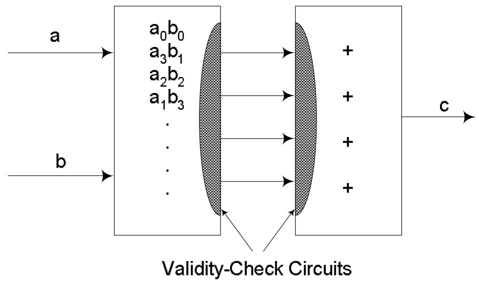

Figure 7: Galois Field Multiplier (Mult 1)

We can combine these two modules by eliminating the validity-check and handshake circuits between them and harboring the functional parts of both modules in a single module. This way we have two back-to-back computational CMOS circuits in a single PCHB-like module that fulfills the limitations of transistors in series. This new circuit, named *Mult 2*, is functionally compatible with the previous one, but has more delay than each of the two parts of *Mult 1*.

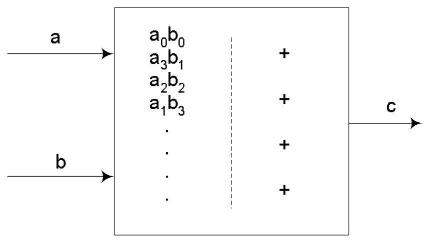

Figure 8: Galois Field Multiplier (Mult 2)

Table 6 shows that by using this method we have reduced the power consumption of GF multiplier by 30% and the number of transistors by 24%.

Table 6: Comparison of two GF multipliers

GF Multipliers	Mult 1	Mult 2
Transistors	1204	912 (**-24.2%**)
Transitions	778	542 (**-30.3%**)

5-5- Internal variables vs. loops

The simplest way to implement a state variable in an asynchronous circuit is to feed an output of a normal element to an input via several buffer stages (Figure 9). Due to the overhead of validity-check circuits, this approach results in a large circuit with excessive power consumption [5, 6].

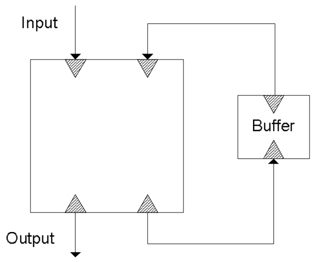

Figure 9: Implementing a state variable by loop

We can implement the states as internal variables to reduce the power consumption and circuit size.

As an example, consider the implementation of the Error Locator part of a Reed-Solomon decoder [11, 14]. This circuit, named Chien Search, has three 4-bit state variables. We have implemented this algorithm using both methods. Table 7 shows the results of these two implementations. The power consumption and circuit size of the second version, implemented as internal variables is 32% and 11% less than those of the first one respectively.

Table 7: comparison of two versions of Chien Search algorithm

Chien	Chien Search with loop	Chien Search with internal variables
Transistors	1352	1200 (**-11.2%**)
Transitions	9931	6762 (**-32%**)

6- Conclusion

We have proposed five methods to reduce the power consumption of asynchronous circuits that consist of PCHB and PCFB templates.

Some of the methods like the third one (Special Circuit Design) cannot be applied to all kinds of circuits and has specific applications, but the others are suitable for most asynchronous circuits.

We have shown that by using these methods we could reach power reduction of up to 40%.

7- References

[1] Al Davis, Steven M. Nowick, “An Introduction to Asynchronous Circuit Design”, *Technical Report UUCS-97-013,* Department of computer Science, University of Utah, September 1997.

[2] J. Sparso, S. Furber, *Principles of Asynchronous Circuit Design – A System Perspective*, Kluwer Academic Publishers, 2002.

[3] C.H.(Kees) van Berkel, Mark B. Josephs and Steven M. Nowick, “Scanning the Technology : Applications of Asynchronous Circuits“, *Proceedings of the IEEE, 87(2):223-233*, 1999.

[4] Alain J. Martin, “Synthesis of Asynchronous VLSI Circuits”,*Caltech, CS-TR-93-28*, 1991.

[5] A. M. Lines. “Pipelined Asynchronous Circuits”, *M.Sc. Thesis, California Institute of Technology*, June 1995, revised 1998.

[6] K. Saleh, “Asynchronous Design Using Pre-Synthesized Templates”, *Technical Report,* Amirkabir University of Technology, September 2003.

[7] K. Saleh, H. Pedram, M. Naderi, M. H. Shafiaabadi, H. Kalantari, A. Farhoodfar, “Synthesis Tool for Asynchronous Circuits Based on PCFB and PCHB”, *Proceedings of the Ninth Annual CSI Computer Conference (CSICC2004),* February 2004.

[8] A. J. Martim, M. Nystrom, K. Papadantonakis, P. I. Penzes, P. Prakash, C. G. Wong, J. Chang, K. S. Ko, B. Lee, E. Ou, J. Pugh, E. V. Talvala, J. T. Tong, A. Tura, “The Lutonium: A Sun-Nanojoule Asynchronous 8051 Microcontroller”, *Proceedings of the Ninth International Symposium on Asynchronous Circuits and Systems (ASYNC'03),* 2003.

[9] A. J. Martin, “Asynchronous Datapaths and the Design of an Asynchronous Adder”, *Formal Methods in System Design, Kluwer, 117-137*, 1992.

[10] D. J. Pursley, S. L. Coumeri, D. E. Thomas, “The Effectiveness of Transition Counting as a Predictor of Relative Energy Consumption”, *Technical Report,* CMUCAD97-43, Department of ECE, CMU, 1997

[11] K. Saleh, “Design and Implementation of an Asynchronous Forward Error Correction Circuit”, *M.Sc. Thesis*, Amirkabir University of Technology, January 2004.

[12] R. O. Ozdag, P. A. Beerel, “High-Speed QDI Asynchronous Pipelines”, *Proceedings of the Eighth International Symposium on Asynchronous Circuits and Systems (ASYNC'02),* 2002.

[13] E. D. Mastrovito, “VLSI Architectures for Computation in Galois Fields”, *PhD thesis*, Linköping University, Department of Electrical Engineering, Sweden, 1991.

[14] K. Saleh, “Hardware Architectures for Reed-Solomon Decoders”, *Technical Report,* Amirkabir University of Technology, January 2003.