ISCAS 2000 - IEEE International Symposium on Circuits and Systems, May 28-31, 2000, Geneva, Switzerland

LOW POWER TRANSFORMATION OF DATAPATH ARCHITECTURES WITH CYCLIC SFGS

Marek Wróblewski [1], *Sven Simon* [2] *and Josef A. Nossek* [1]

[1] Munich University of Technology
Arcisstr. 21, 80333 Munich, Germany
Marek.Wroblewski@ei.tum.de

[2]Infineon Technologies
St.-Martin-Str. 76, 81541 Munich, Germany
Sven.Simon@infineon.com

ABSTRACT

Datapath architectures exhibit a large amount of undesired switching (glitches) which does not contribute to the functionality but leads to increased power consumption. While glitch propagation can be effectively reduced by pipelining circuits with acyclic SFGs, this technique is not directly applicable if the circuit contains loops. The paper addresses this problem and discusses a methodology for reducing switching activity in recursive circuits. Simulation results of a few example circuits follow.

1. INTRODUCTION

The generous power budget required by CMOS circuits has become a major concern in recent years. Continuosly increasing performance and functionality requirements leading to more complex circuits and higher clock rates make the power specifications soar. Simultaneously the demand for mobile systems and low cost solutions dispensing with expensive cooling measures makes it necessary to minimize energy expenditure.

An important class of low-power methodologies are based on minimizing glitch propagation in combinatorial circuits using glitch barriers by signal gating or by additional registers. In [3] pipeline registers are inserted in order to reduce glitch propagation in a multiplier array. In [7] logic gates and latches are employed for signal gating. This latter approach requires however a careful design of control signals and is in particular difficult to pursue if the design effort is shared by several independent teams. In contrast to this, synchronously clocked registers are widely supported in the standard design flow by fairly well estabilished techniques of clock tree synthesis and static timing analysis. Therefore in order to facilitate the integration of low power methodologies into the standard design flow we consider this approach advantageous.

In the following, we discuss a procedure leading to a decreased switching activity in circuits where pipelining is not applicable because the signal flow graph (SFG) contains cycles. We use slowdown [2] and retiming in order to achieve a register distribution that results in reduced glitching and power consumption.

2. TRANSFORMATION OF THE CIRCUIT

We use the following conventions from [2]: We model a digital synchronous circuit as a directed connected multigraph G, with vertices representing combinatorial blocks of the circuit and edges representng connections between them. Let E be the set of edges and V be the set of vertices of G. Each edge $e(u,v) \in E$ from vertex u to vertex v has a number of registers $w(u,v)$ associated with it and every node is weighted with its propagation delay d.

Thus, we write $G =< E, V, w, d >$.

A directed path p is an ordered set of edges $e_i(u,v) \in G$ such that
$p = \{\ldots, e_i(u,v), e_{i+1}(v,w), \ldots\}, i = 0, 1, \ldots, |p| - 1$.
We define P as a set of all directed paths p of G such that:

- $e_0(u,v) \in p \Leftrightarrow u$ is a primary input,
- $\exists e_i(u,v), e_j(w,u) \in p, j \neq i-1 \Leftrightarrow$ $j = |p| - 1$.

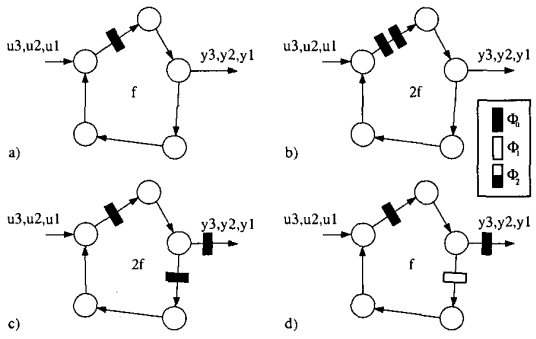

Figure 1: The transformation of an SFG for power reduction.

Starting from the original circuit G we perform the following steps to obtain a low power optimized architecture of G (Fig. 1 illustrates the procedure by an example):

1. Apply slowdown $\mathcal{S}$ to G to obtain the m-slow form of G:

   $$G_m =< E, V, w_m, d >= \mathcal{S}(G),$$

   with $w_m(u,v) = m \cdot w(u,v)$.

2. Apply retiming $\mathcal{R}$ to G_m in order to redistribute the registers in the circuit

   $$G_{mr} =< E, V, w_{mr}, d >= \mathcal{R}(G_n).$$

   The goal of this transformation is to find a register distribution for which the switching power in the circuit is minimal. In general this is not equivalent to positioning registers in edges with the highest switching activity (cf. Fig. 2). In our design examples we chose edges with the highest "switching activity–capacitive load" product.

3. Introduce an m-phase clocking scheme. Assign registers a clock phase i, $i \in \{0, \ldots, m-1\}$.

   In order to assign clock phases to registers the following procedure can be executed:

   For each $p \in P$ perform the following:

   (a) set $k = 0$.

   (b) Starting in the first edge of p traverse the path to its end. For every register encountered assign it number k and increment k by one.

   Assign the register k the clock phase $(k \mod m)$. In section 3 we show that the above scheme introduces no contradictions.

4. Consider an edge containing more than one register. Let the last register of this edge be assigned number k. Then all registers which are not triggered by the clock phase $(k \mod m)$ can be removed from this edge.

3. ASSIGNMENT OF CLOCK PHASES

Let us discuss the above procedure of assigning clock phases. Let

$$T = \{(p,q) | p, q \in P,\ \exists e \in E : e \in p \wedge e \in q\}.$$

We want to show that executing this procedure cannot lead to a situation where the phase assigned to a register while traversing a path p differs from the phase assigned to it while traversing a path q, i.e. that

$$\forall (p,q) \in T : \kappa_p \mod m = \kappa_q \mod m,$$

where κ_i denotes the value of k at the starting node of the first common edge of p and q (node α in Fig. 3), as obtained by traversing the path $i, i \in \{p,q\}$.

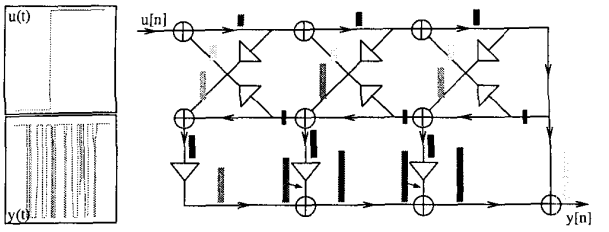

Figure 2: Glitching in a sample circuit. The height of bars indicates the glitch count at a given edge, while the grayscale represents dissipated energy (darker means more dissipation). $u(t)$ and $y(t)$ show signal waveforms at circuit's input and ouput, respectively.

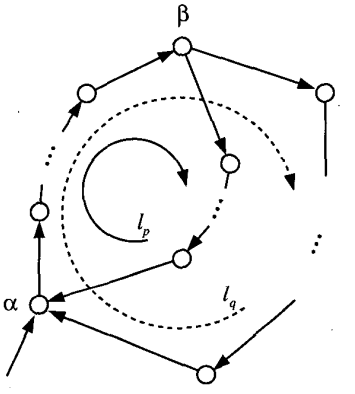

Figure 3: The general case for discussion of clock phase assignment.

To avoid case differentiation, we introduce an additional dummy node δ with delay 0, and edges going from δ to each primary input and from each primary output to δ. Then every path $p \in P$ contains a loop, i.e.

$$\forall p \in P\ \exists e_i(u,v), e_j(w,u) \in p : j \neq i-1.$$

The discussion can then be reduced to the scenario shown in Fig. 3.

We consider paths p, q such that $(p,q) \in T$. Let the loop $l_p, l_p \subseteq p$, have $N_p = n_p$, and the loop $l_q, l_q \subseteq q$, have $N_q = n_q$ registers before the application of slowdown. For

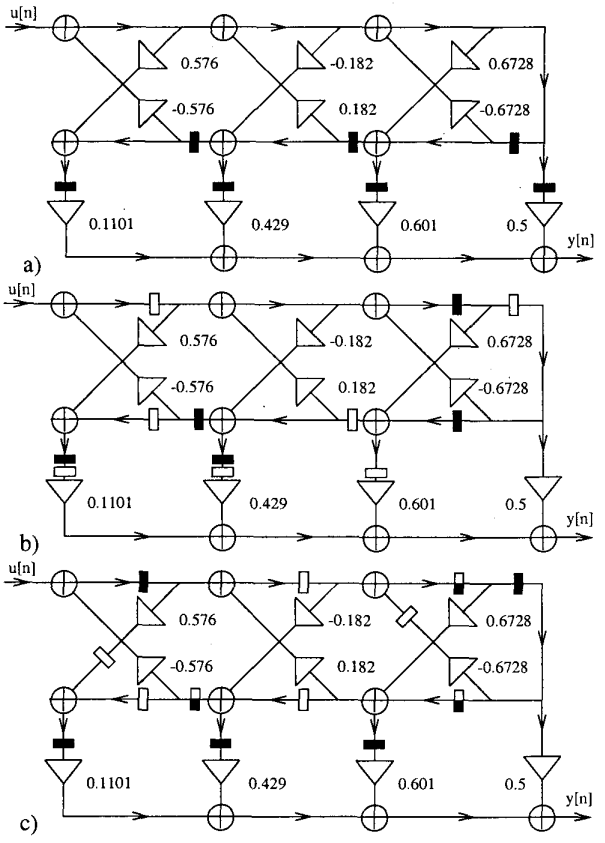

Figure 4: Example 1 – The original circuit (a) and two low power optimized architectures (b and c).

the m-slow form holds then

$$N_p = m \cdot n_p, \quad N_q = m \cdot n_q. \tag{1}$$

Retiming does not change the amount of registers in a loop [2], so the Eqs. 1 still hold. Let the path between nodes α and β contain $m \cdot n_c + c$ registers, $n_c \in \{0, \ldots, \min(n_p, n_q)\}$, $c \in \{0, \ldots, m-1\}$. Let us also assume, without loss of generality, that at the node β holds: $k \mod m = 0$, i.e. that the first register found after this node will be assigned phase 0. Then

$$\kappa_p = m \cdot n_p - m \cdot n_c - c = m \cdot (n_p - n_c - 1) + m - c$$

$$\kappa_q = m \cdot n_q - m \cdot n_c - c = m \cdot (n_q - n_c - 1) + m - c$$

and

$$\kappa_p \mod m = \kappa_q \mod m = \begin{cases} 0 & \text{if } c = 0, \\ m - c & \text{otherwise.} \end{cases}$$

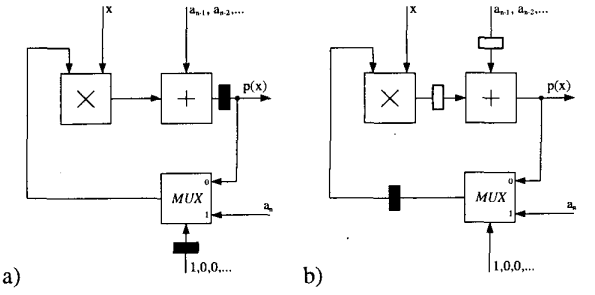

Figure 5: Example 2 – A recursive structure for polynomial evaluation ($p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$).

4. IMPLEMENTATION AND RESULTS

In order to verify the effectiveness of the methodology, we investigated several circuits containing loops and designed their low-power versions following the above described flow.

The circuits were described in VHDL and synthesized employing Synopsys' DC Compiler. Power estimation was performed using Powrmill.

The results are presented in Table 1.

Fig. 4 shows a lattice filter. Power consumption of this ciruit can be significantly reduced by positioning registers in edges with high capacitive load (cf. Fig. 4b). The introduction of a 3-phase clocking scheme leads to even more interesting power figures.

Fig. 5 shows two architectures of a recursive structure for evaluation of polynomials [5], [1]. The cost of additional clocking circuitry in these circuits is almost as high as the savings due to reduced glitching, thus allowing only a modest power reduction.

In contrast, the modification of the lattice filter of Fig. 6 [4] yields remarkable savings in spite of using as many as three clock phases.

5. CONCLUSIONS

In this paper we have discussed a slowdown-based methodology for reduction of power consumption in circuits with cyclic SFGs by using glitch barriers.

The application of the method puts the engineer in a classical trade-off situation where the cost of additional circuitry needed to construct glitch barriers has to be outweighed by savings achieved due to reduced glitching. The examination of a few example datapath circuits shows that the method can be of great advantage in designs containing long combinatorial paths composed of gates with high fanout. The register distribution in the presented circuits was obtained by application of simple heuristics and the results may further be improved if a more sophisticated approach is used.

Arch	P_c	P_s	P_{tot}	Δ	P_c	P_s	P_{tot}	Δ	P_c	P_s	P_{tot}	Δ
	8bit				16bit				32bit			
Circuit of Fig. 4												
a	8.02	0.565	8.58	-	26.26	0.715	26.98	-	65.555	1.03	66.585	-
b	5.55	1.135	6.685	22.13	18.005	1.53	19.535	27.58	43.905	2.365	46.27	30.51
c	4.665	1.565	6.225	27.45	14.055	1.94	15.995	40.72	32.92	2.76	35.68	46.41
Circuit of Fig. 5												
a	0.855	0.45	1.305	-	2.2	0.46	2.66	-	12.105	0.525	12.625	-
b	0.725	0.52	1.245	4.66	1.92	0.565	2.48	6.71	10.985	0.81	11.795	6.57
Circuit of Fig. 6												
a	27.61	0.52	28.13	-	120.835	0.66	121.495	-	362.81	0.955	363.765	-
b	12.18	1.83	14.01	50.19	32.555	2.275	34.83	71.33	80.565	3.435	84.005	76.91

Table 1: Simulation results. P_c denotes power consumed by combinatorial parts, P_s – power consumed by clock buffers and registers. $P_{total} = P_c + P_s$, $\Delta = \frac{P_{total,a} - P_{total,i}}{P_{total,a}} \cdot 100\%$, $i \in \{a, b, c\}$. All power figures are in mW.

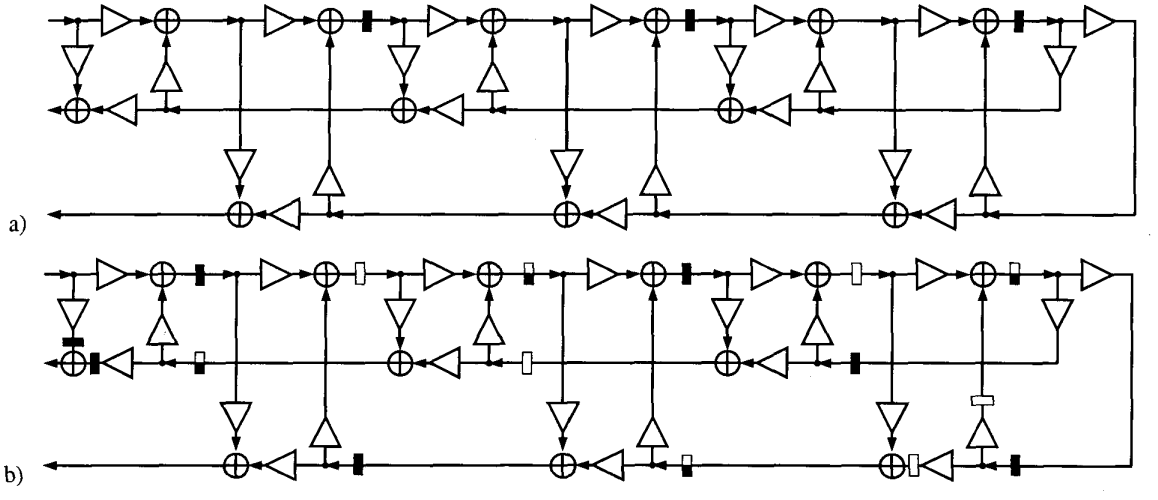

Figure 6: Example 3 – Original digital filter from [4] (a) and its modified version (b).

Future work will focus on formulating the retiming problem in a more stringent way in order to find the optimal register positions and explore the trade-off between the number of clock phases and total power consumption of the circuit. This represents the first step towards the automation of the methodology and its integration in the semi-custom design flow.

6. REFERENCES

[1] K. Hwang. *Computer Arithmetic: Principles, Architecture, and Design*. John Wiley & Sons, 1979.

[2] C. E. Leiserson and J. B. Saxe. Retiming synchronous circuitry. *Algorithmica*, pages 5–35, 1991.

[3] J. Monteiro and S. Devadas. *Computer-Aided Design Techniques for Low Power Sequential Logic Circuits*. Kluwer Academic Publishers, 1997.

[4] S. K. Rao and T. Kailath. Orthogonal digital filters for VLSI implementation. *IEEE Transactions on Circuits and Systems*, CAS-31(11):933–945, November 1984.

[5] N. R. Scott. *Computer Number Systems and Arithmetic*. Prentice Hall, 1985.

[6] S. Simon and M. Wróblewski. Low power datapath design using transformations similar to temporal localization of SFGs. In *Proc. IEEE Int. Symp. on Circuits and Systems*, Orlando, USA, May 1999.

[7] G. K. Yeap. *Practical Low Power VLSI Design*. Kluwer Academic Publishers, 1998.