Retiming of Synchronous Circuits with Variable Topology

Sven Simon, Ralf Bucher* and Josef A. Nossek
Institute for Network Theory and Circuit Design
Technical University Munich
Arcisstr. 21, 80333 Munich, Germany
svsi@nws.e-technik.tu-muenchen.de

Abstract

Generally, circuit design leads to a trade-off scenario between speed and various parameters like power dissipation, AT complexity, re-use of already existing cells, design time, etc. To deal with this trade-off, the interaction between retiming and the selection of combinational elements from a set of cells with these different parameters is considered here. Additionally, modifications of the circuit graph concerning the interconnections, e.g. due to associativity of the underlying algorithm, lead to a parameterized topology. The algorithm presented in this paper combines all three, retiming, the selection of specific cells and the choice of an appropriate topology in one optimization step.

1 Introduction

Retiming has long been a technique of great interest to the field of CAD research and development. In order to maximize circuit performance registers are inserted and deleted in a way that data dependencies remain unchanged. Unlike pipelining, this technique does not change circuit latency. Retiming can be applied manually using a cut-set transfer rule, see e.g. [1]. Leiserson, Rose and Saxe [8] presented polynomial algorithms to find an optimal solution for retiming concerning maximum clock speed, minimum number of registers, and minimum cost, if cost is assigned to each possible register location.

Another field of considerable interest, which plays the major role here, is the combination of retiming with circuit modification. De Micheli examines logic synthesis and cycle-time minimization without separating the combinational elements from registers, see [2]. In [3], Malik *et al.* consider retiming with logic synthesis. Registers are temporarily removed from the circuit in order to apply combinational optimization to the logic elements. Additionally in [4, 5], Potkonjak, Dey *et al.* apply algebraic speed up to those temporarily register free subcircuits.

Unlike their approach, the methodology presented in this paper merges the interaction between retiming and the modification of the circuit graph into one global performance optimization to make sure that the global optimum is achieved. Edges belonging to different topologies are allowed to contain arbitrary register numbers during the change of the topology. In addition to architectural alternatives, the minimization of a cost-function depending on the cell based parameters mentioned above can be used in order to find the most efficient circuit. Thus, the optimum AT[1] complexity, maximum efficiency $\eta = f/A$ [7] or minimum power dissipation for each required clock period can be evaluated. The optimization of these design measures, meets practical design strategies, e.g. see [1, 7]. This methodology is intended, but not limited to full-custom design.

The same graph model and circuit, presented in Leiserson's paper, is used for further discussion, see Fig. 1. A circuit G is characterized by the graph (V, E), the edge weights w and delays d. This can be abbreviated by $G=< V, E, d, w >$. Each vertex of Fig. 1 represents a functional element, containing no timing element, neither a register nor a latch. The worst case of data propagation delay $d(v)$ is associated with each vertex $v \in V$. The paths for data transfer are introduced as directed edges $e(u \rightarrow v) \in E$ from vertex u to v. The in- and outdegree of every vertex is greater than or equal to one. The number of registers of an edge e is defined as the edge weight $w(e)$ and the number of registers moved over every vertex v by retiming is assigned to the variable $r(v)$. Thus, the

*R. Bucher is now with the Semiconductor Group, Siemens AG, Munich

[1] Measures like AT complexity or AT^2 are known from comparison of implementations of different algorithms [1, 6]. They can be used similarly to choose among a set of implementations based on different cells and interconnections.

1063-9667/95 $4.00 © 1995 IEEE

modified edge weight w_r of an edge $e(u \rightarrow v)$ after retiming is:

$$w_r(e) = w(e) + r(v) - r(u). \qquad (1)$$

The example, originally presented in Leiserson's paper, is slightly modified by an additional register between the vertices with delay 0 and 3. Therefore, another optimum speed retiming solution, compared to [8] is obtained, see Fig. 1.

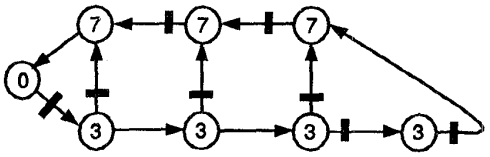

Fig. 1. The optimum speed retiming solution, with T_{min}=9.

Up to now, only one circuit graph G is regarded for retiming. The combinational element with delay 7 in Fig. 1 originates from a full-adder [8]. Let us assume that different adder cells with delays given in Table 1 are available for implementation. Another maximum speed solution is obtained, if a faster cell is chosen. It is reasonable to assume that this cell requires more area.To meet the design strategy of optimum efficiency mentioned above it suggests itself to minimize an area cost-function applying retiming combined with cell selection.

In addition to the variable node delays, the edge structure of the circuit can be changed due to the associativity of addition. According to cell abutment known from full-custom and semi-custom design the original design needs an additional data bus, if floorplanning leads to a separated input output interconnection as shown in Fig. 2. In order to implement the correlator efficiently, a retiming procedure is applied, which considers these topological and cell based alternatives and evaluates the optimum circuit concerning a cost-function.

In the following section, an algorithm, which is derived form Leiserson's mixed-integer linear-programming (MILP) of retiming is presented. In Section 3, we discuss how to apply different design measures to the cost-function. Finally, in Section 4, the derived algorithm is applied to the correlator and the optimum circuit for each possible period is computed.

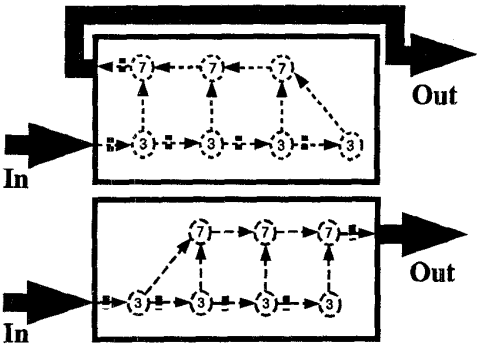

Fig. 2. Floorplanning of two versions of the correlator.

2 The algorithm

In this section, we modify the mixed-integer linear-programming formulation (MILP) of retiming presented in [8].

An algorithm to solve this special MILP in $O(|V||E| + |V||V|lg|V|)$ is given in [9]. Although we sacrifice the prerequisite for this time bound due to our modifications, a comparable polynomial order can be maintained. A detailed discussion and experimental results are given in Section 4.

2.1 Variable propagation delays

In order to formulate retiming as a MILP problem Lemma 9 from [8], denoted here as Lemma 1, is used.

Lemma 1. A synchronous circuit G satisfies a clock period c by retiming, if the following conditions hold for a real unknown $s(v)$ of each vertex v.

1.1. $-s(v)/c \leq -d(v)/c$
for every vertex v $\in V$

1.2. $s(v)/c \leq 1$
for every vertex v $\in V$

1.3. $w_r(e) \geq 0$
for every edge $e(u \rightarrow v)$

1.4. $s(u)/c - s(v)/c \leq -d(v)/c$
for every edge $e(u \rightarrow v)$ and $w_r(e) = 0$.

Condition 1.4. is not suited for a MILP formulation, because $w_r(e)$ varies during retiming. To solve this problem an auxiliary variable z is added to the right side of Equation (1.4).

$$s(u)/c - s(v)/c \leq z - d(v)/c. \qquad (2)$$

This variable controls, whether (2) constrains the unknown s or not. Obviously, if $z = 0$, Condition 1.4 holds. If $z \geq 1$, (2) provides no constraint on s. To make this clear (2) can be written slightly different:

$$s(u)/c - z \leq s(v)/c - d(v)/c. \quad (3)$$

If $z \geq 1$, the right side of (2) is positive because of (1.1) and the left side is negative because of (1.2) for every s.

If z is replaced by $w_r(e)$, the condition 'and $w_r(e) = 0$' in (1.4) can be omitted, because $w_r(e)$ controls (2) inherently. Thus, a MILP formulation of Lemma 1 can be obtained, which is identical to those in [8]. In order to integrate a choice of different cells we have to consider both, the propagation delays and the cell based parameters. The delays can be introduced using boolean variables $x_\alpha(v)$ for every vertex v with $x_\alpha(v) = 1$ whenever cell α is chosen and zero otherwise. The delay $d(v)$ of a vertex v with n different implementations can be written as follows:

$$d(v) = \sum_{\alpha=1}^{n} d_\alpha(v)x_\alpha(v); \quad \sum_{\alpha=1}^{n} x_\alpha(v) = 1; \quad x_\alpha(v) \in \{0,1\}. \quad (4)$$

In order to choose the most suited cell, we have to analyze the area, power dissipation or to estimate the design time, if cells do not already exist. This can be achieved, if the values $a_\alpha(v)$ of those parameters are used to formulate a cost-function $f_c = \sum_v \sum_\alpha a_\alpha(v)x_\alpha(v)$, as usually done for MILP problems.

2.2 Variable interconnections

Similar to subsidiary point a, the variable z is used to introduce variable interconnections. First, an estimation about the maximum number of registers in an edge e is needed.

Lemma 2. Let G be a synchronous circuit and let the matrix W contain the minimum number of registers of any path from u to v in its entry $W(u,v)$ [8]. The maximum number of registers $w_{max}(e)$ of an edge $e(u \rightarrow v)$ is:

$$w_{max}(e(u \rightarrow v)) \leq W(u,v) + W^T(u,v). \quad (5)$$

Proof. The total number of registers of a directed cycle is invariant during retiming [8, 10]. An entry in the matrix $W^T(u,v)$ contains the number of registers on a path $p(v \rightarrow u)$. Thus, $w_m = W(u,v) + W^T(u,v)$ is the number of registers on a directed cycle containing the edge $e(v \rightarrow u)$. Therefore, the maximum number of registers in e is less or equal to the total amount in the cycle: $w_{max} \leq w_m$. □

Let w_M be the maximum number w_m of all edges in all topologies. This can be computed in $O(T|V|^3)$, if T is the number of different topologies, because W can be evaluated applying the Floyd-Warshall algorithm $O(|V|^3)$ [8].

In order to deal with different topologies a set of boolean valued variables y_i is needed with $y_i = 1$, if topology i is chosen and zero otherwise. Every edge $e(u \rightarrow v)$ belongs to one or a set of topologies. Thus, the subscript i for each edge $e_i(u \rightarrow v)$ is used to identify the topology. Insert $z = w_r + (w_M + 1)(1 - y_i)$ and $s(v) = c(R(v) - r(v))$ in Lemma 1 with Condition 1.4 replaced by (2), see Theorem 3.

Theorem 3. A synchronous circuit G satisfies a clock period c, if there exist values for the real variable $R(v)$ and integer variable $r(v)$ such that conditions 3.1-3.6 hold. The optimum circuit, which fulfills these conditions, is defined as the circuit with the global minimum of the cost-function $f_c = \sum_v \sum_\alpha a_\alpha(v)x_\alpha(v) + \sum_i b_i y_i; a_\alpha(v), b_i \in \mathrm{R}$.

3.1. $r(v) - R(v) \leq -\sum_\alpha x_\alpha(v)d_\alpha(v)/c$
for every vertex v $\in V$

3.2. $R(v) - r(v) \leq 1$
for every vertex v $\in V$

3.3. $r(u) - r(v) \leq w(e_i) + w_M(1 - y_i)$
for every edge $e_i(u \rightarrow v)$

3.4. $R(u) - R(v) \leq$
$w(e_i) + (w_M + 1)(1 - y_i) - \sum_\alpha x_\alpha(v)d_\alpha(v)/c$
for every edge $e_i(u \rightarrow v)$

3.5. $\sum_\alpha x_\alpha(v) = 1; \quad x_\alpha(v) \in \{0,1\}$
for every vertex v $\in V$

3.6. $\sum_i y_i = 1; \quad y_i \in \{0,1\}$

Proof. The introduction of a variable delay is correct from (4). Condition 3.3 allows arbitrary negative edge weights w_r, if an edge does not belong to the chosen topology. This is necessary, because all possible retiming solutions of another topology must be permitted. For a selected topology condition 3.3 changes to $w_r \geq 0$, which corresponds to 1.3 of Lemma 1. Similarly, condition 3.4 is only allowed to constrain the variable $r(u)$ and $r(v)$, if the considered edge does not belong to the chosen topology. This is because the inserted value of z is at least 1, if another topology is chosen. Due to the discussion about z Condition 3.4 does not constrain the variables $r(u)$ and $r(v)$. Condition 3.5 and 3.6 enforce that exactly one cell and

topology are selected. The cost-function f_c depends on both, the cell variables x_α and topology variables y_i. □

3 Design measures

In order to examine different aspects of design, several cost-functions have to be dealt with. As MILP formulations only allows one function for optimization, we have to combine them to a common cost-function. The following two cases have to be considered.

Firstly, if two cost functions f_{c1} and f_{c2} are correlated or dependent on the same variables, but represent different design measures, the priority of one can be chosen during optimization. The MILP problem will optimize f_{c1} first, if the following cost-function is used: $f_c = k f_{c1} + f_{c2}; \quad k > |\Delta_{max}(f_{c2})|$.

Secondly, if the variables of both cost-functions f_{c1} and f_{c2} are not correlated, both are minimized if the sum $f_c = f_{c1} + f_{c2}$ is minimized.

Thus, we can merge the following design measures into one cost-function.

Area. Area consumption is influenced by both, replacement of registers and the selection of different combinational cells. The variables are not correlated so that we can use the sum of both cost-functions.

Power Dissipation. In order to minimize the power dissipation of the final circuit, a cost function containing the power dissipation of the cells has to be formulated. These values are known from a library or can be found by simulation.

Design Time. We can either re-use already existing cells for a new layout or design new cells. If older cells have to be modified, an estimation for this change can be used. Similarly, an estimation to design a complete new cell can be used.

Efficiency. The optimum Efficiency η, e.g. used in [7], can be evaluated, if a cost-function to minimize the total amount of area is used. With this method the efficiency or AT complexity for each possible clock period can be computed.

4 Experimental results

In this section the results applying the MILP formulation of Theorem 3 to the correlator is presented. In order to deal with a realistic problem we choose values for area and propagation delays from a full-custom library, see e.g. [11, 12].

Cell	Delay	Area	Schematic
full-adder a	3	7.2	unsymmetric f.a.
full-adder b	5.7	3.1	symmetric f.a.
full-adder c	2.5	10.2	f.a. with T.-Gate
xor	1.8		only one type

Table 1: Propagation delays and areas of different full-adder (f.a.) types (T.-Gate = Transmission-Gate).

Generally, high-throughput implementations can be achieved applying carry-save (CS-) arithmetic to adder chains. Using CS-arithmetic for the correlator the propagation delay of one adder is identical to that of a full-adder given in Table 1. Three different cells concerning propagation delay and area are assumed being available to implement a full-adder. These adders result from different schematics in CMOS design. Additionally, two topological alternatives are obtained, if the associativity mentioned in the introduction is applied. The topology of the second circuit of Fig. 2, denoted as Type II, should be preferred to Type I, which is the usual correlator. In order to give an MILP formulation, we have to divide interconnections e into three different sets e_I, e_{II} and e_{III}, see Fig. 3. The first two contain the edges of Architecture-Type I and II, whereas edges used for both architectures belong to the third set.

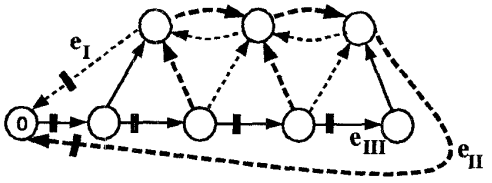

Fig. 3. Three sets of edges e_I, e_{II} and e_{III} covering all architectural alternatives.

As mentioned in Section 3 two possible priorities to implement a cost-function can be chosen for minimization $f_{CA} = -y_I - 2y_{II} + \sum_\alpha A_\alpha x_\alpha;\ |A_\alpha - A_\beta| > 1$ and $f_{CB} = -10y_I - 20y_{II} + \sum_\alpha A_\alpha x_\alpha;\ |A_\alpha - A_\beta| < 10$.

A program [13] is used to solve general MILP problems for this instance of Theorem 3. Additionally, we introduced modularity enforcing equal register numbers in edges, which belong to identical functional elements. For the correlator modularity is applied to the edges between the xor's. In Table 2 the optimum solution for each possible clock period is presented for the correlator with and without modularity. This Table enables the designer to choose the 'best' circuit for a certain range of a required clock period.

The average computation time of the 116 MILP problems for all possible clock periods of this example is 48 seconds (sparc 2). Even though MILP problems being not polynomial in general, Leiserson presented an algorithm $O(|V||E|+|V||V|lg|V|)$ to solve the specific retiming problem. Although the prerequisites, which lead to a polynomial bound in Leiserson's paper, is sacrified the number of different circuits due to variable propagation delays and topologies does not increase, if the length of the correlator rises with $|V|$. If the total number S of circuits is invariant concerning arbitrary values of $|V|$, the computation time is $O(S|V||E|+S|V||V|lg|V|)$. The variable S is generally limited to a value independent of $|V|$, if it is assumed that full-custom design is usually accompanied by a hierarchical and modular design style.

Clock Period	No Modularity					
	Cost:f_{CA}			Cost:f_{CB}		
T≥4.3	Cell	Type	f_{CA}	Cell	Type	f_{CB}
-4.8	c	I	9.2	c	I	0.2
-5.0	a	I	6.2	a	I	-2.8
-5.7	a	I	6.2	c	II	-9.8
-6.0	b	I	2.1	c	II	-9.8
-10.8	b	I	2.1	a	II	-12.8
-18.9	b	II	1.1	b	II	-16.9
Modularity						
-5.0	-	-	-	-	-	-
-6.0	c	II	8.2	c	II	-9.8
-7.2	a	II	5.2	a	II	-12.8
-10.8	b	I	2.1	a	II	-12.8
-18.9	b	II	1.1	b	II	-16.9

Table 2: Solutions of Theorem 3 for all possible clock periods.

5 Conclusion

In this paper, Leiserson's mixed-integer linear-programming formulation of retiming is extended so that topological alternatives can be combined with retiming. Two classes of topological alternatives are covered, the choice of the most efficient cell out of a set of implementations for a functional unit and the selection of the most suitable edge structure of alternative interconnections. Due to the combination of these topological variations with retiming an optimum circuit concerning different design measures like efficiency, minimum area or power dissipation, etc. can be found for a required clock period.

The methodology, presented here, can be used to support a designers choice of different design alternatives. A designer can combine layout information (aread, delay, cells, etc.) with high level architectural aspects in order to obtain an optimized circuit. Future work will focus on an efficient implementation of this algorithm in an interactive design automation system.

Acknowledgements

I would like to thank Dr. Schoebinger, W. Kamp and H. Soeldner (R & D, Siemens AG, Munich) for fruitful discussions about design strategies and their experiences.

References

[1] S. Y. Kung: VLSI Array Processors. Englewood Cliffs, Prentice-Hall, 1988.

[2] G. De Micheli: Synchronous Logic Synthesis: Algorithms for Cycle-Time Minimization . IEEE Transactions on Computer-Aided Design **10** (1991), 63–73.

[3] Malik, S.; Sentovich, E. M.: Optimizing Sequential Networks with Combinatorical Techniques . IEEE Transactions on Computer-Aided Design **10** (1991), 74–84.

[4] M. Potkonjak, S. Dey, Z. I.; Parker, A. C.: High Performance Embedded System Optimization Using Algebraic and Generalized Retiming Techniques. ICCD 93, 1993.

[5] S. Dey, M. P.; Rothweiler, S. G.: Performance Optimization of Sequential Circuits by Eliminating Retiming Bottlenecks. ICCAD, 1992. 510–517.

[6] J. D. Ullman: Computational Aspects of VLSI. Computer Science Press, Rockville, 1984.

[7] T. G. Noll: Carry-Save Architectures for High-Speed Digital Signal Processing. Journal of VLSI Signal Processing (1991), 121–140.

[8] C. E. Leiserson and J. B. Saxe: Retiming Synchronous Circuitry . Algorithmica (1991), 5–35.

[9] C. E. Leiserson and J. B. Saxe: A Mixed-Integer Linear Programming Problem Which Is Efficently Solvable. Journal of Algorithms **9** (1986), 114–128.

[10] S. Simon, E. Bernard, M. Sauer and J. A. Nossek: A New Retiming Algorithm for Circuit Design . Proceedings of the International Symposium on Circuits and Systems, ISCAS, to appear, 1994.

[11] D. V. Heinbuch: CMOS Cell Library. Fort Meade, Maryland, Addison-Wesley, 1988.

[12] H. Klar and W. Heimsch: Integrierte Digitale Schaltungen MOS/BICMOS. Munich, Springer-Verlag, 1993.

[13] M.R.C.M. Berkelaar: Solve Mixed Integer Linear Programming Problem. Design Automation Section, Eindhoven University of Technology, 1993.