An Extensive Survey On Efficient Retiming of Fixed-Point Circuits

Kshipra Tiwari[1], Prof. Sharad Mohan Shrivastava [2]

[1]MTech. Scholar, [2]Research Guide

Department of Electronics and Communication Engineering, SISTEC, Bhopal

***Abstract-* The rapid growth of semiconductor technology has fuelled large scale VLSI innovation in the electronic product design domain. The electronic gadgets available in the market today are not only useful in their own right but there is intense competition between various industries for making them efficient, multi-functional and less expensive. In order to keep up with the increasing competition, the product design engineers find themselves in a perpetual cycle of design and product delivery. With every iteration of the cycle, the designers need to work with increased technological complexity while aim to deliver more value for every unit cost that charge from the consumer. The present research addresses the problem of assigning optimal fixed-point number formats to operations for the implementation of a given signal processing system, graphic processing system etc. keeping in mind both correct functionality and being able to meet the desired performance parameters.**

***Keywords-* Cutset retiming, digital signal processing (DSP) hardware, fixed-point arithmetic, retiming, GPU, FFT.**

I. INTRODUCTION

With the advances in integrated circuit (IC) technology, more than 10 million devices can be manufactured on a single chip today. Because of this increase in the complexity, Very Large Scale Integration (VLSI) circuit designs require sophisticated Electronic Design Automation (EDA) tools capable of handling large circuits. Due to the increase in complexity and reduced time to market, designers cannot rely on their intuition to design fast, low power sequential circuits with minimum area. Thus circuit optimization tools are indispensable for designers, and much work needs to be done to develop good computer-aided design (CAD) tools. Most of the traditional circuit optimization techniques operate on combinational sub-circuits extracted from sequential designs. Thus to limited capabilities for optimization and true sequential optimization techniques are needed. This work develops CAD tools for optimizing large sequential circuits.

Retiming is a powerful transformation that has great potential for sequential circuit optimization. It is the concept of moving storage devices across computation nodes to improve performance without changing the input-output behavior, and can operate at gate level netlists or higher abstractions (e.g. data flow graphs, communication graphs, processor schedules).

At the circuit level these storage devices are called registers which can be either edge- triggered flip-flops (or FF's) or level sensitive latches (or latches), and the computation nodes are combinational gates. Retiming moves registers across gates without changing the number of registers in any cycle or on any path from the primary inputs to the primary outputs. This preserves the input-output latency of the circuit. Since retiming does not directly affect the combinational part of the circuit the circuit behavior remains unchanged. However since retiming can change the boundaries of combinational logic, it has the potential to affect the results of combinational synthesis as well.

Retiming can be performed to improve the circuit behavior with respect to different objective functions. Some of these objective functions are discussed below.

A. Clock Period

The simplest objective function used in retiming is minimization of the clock period. Since the clock period in an edge-triggered circuit is given by the maximum combinational delay, registers can be relocated to reduce the clock period.

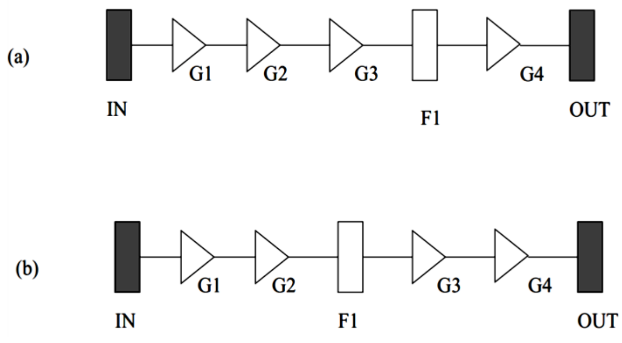

Figure 1.1 retiming effects on clock period.

For the circuit shown in Figure 1.1 (a), with unit delay gates, the clock period is 3.0 time units. If relocate register LI from the output of gate G3 to its input, get the circuit in Figure 1.1 (b). with a clock period of 2.0 units. Thus relocating registers can reduce the clock period of a circuit, and retiming can be used to minimize the clock period.

Retiming to minimize the clock period is termed minperiod retiming. Notice that the input-output behavior is not changed by retiming since in both cases the output is produced after 2 clock cycles. Retiming a circuit to achieve a given target clock period is a special case of this problem.

B. Area

Since retiming does not affect the combinational part of the circuit, the area overhead of the combinational part remains constant. Retiming can. However, affect the overall area of the circuit since it can alter the number of registers in the circuit.

Two circuits can have the same input-output behavior and clock period, but require different number of registers. To illustrate this consider the circuits in Figure 1.2 which are equivalent under the retiming transformation. The circuit in Figure 1.2 (a) requires two registers while that in Figure 1.2 (b) requires only one register.

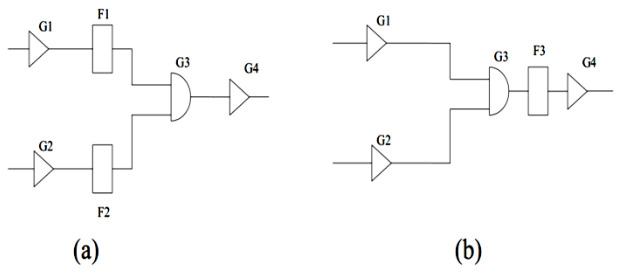

Figure 1.2 Effect of retiming on number of registers.

Retiming can therefore be used to minimize the number of registers in the circuit. This can be done without any constraint on the clock period of the resulting circuit, or subject to a target clock period. The former is called unconstrained min area retiming while the latter is called constrained min area retiming or simply min area retiming.

C. Power

The power dissipated in a circuit depends on the product of switching activity and the load capacitance at the output of a gate, summed over all gates. Since registers can filter out glitches, relocation of registers will affect the switching activity at gate outputs. In addition relocating registers also changes the load capacitance seen by gates. Thus retiming can change the power requirements of a circuit, and can be used for reducing the power dissipation in sequential circuits by placing registers on interconnections with high switching activity and high capacitive loads.

II. RETIMING DATA FLOW GRAPHS

In data-flow-graph representations, the nodes represent computations (functions or subtasks) and the directed edges represent data paths (communication between nodes).

Retiming changes the location of registers (delay elements) in a circuit in an attempt to balance the logic depth between sequential elements and minimize the critical path. A valid retiming solution must not change the input/output functionality of the DFG.

Retiming is mainly used to reduce the critical path in synchronous circuits. The critical path of the filter in Fig. 2.1(a) (shown by the dashed line) passes through one multiplier and one adder and has a computation time of 3 u.t. The retimed filter in Fig. 2.1(b) has a critical path that passes through two adders and has a computation time of 2 u.t.

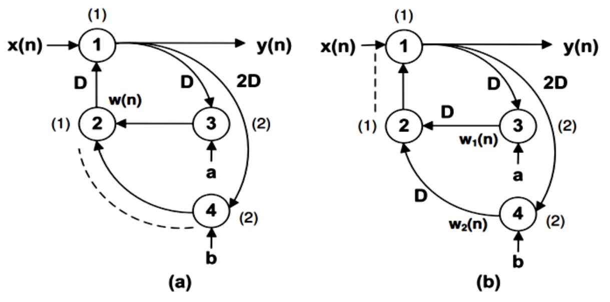

Figure 2.1 (a) Original DFG (b) Retimed DFG.

A. Mathematical Model for Retiming

Retiming maps a data-flow-graph G to a retimed graph G_r. A retimed solution is characterized by a value r(U) known as the retiming weight for each node U in the graph. Let $\omega(e)$ denote the weight of the edge e in the original graph G, and let $\omega_r(e)$ denote the weight of the edge e in the retimed graph G_r. The weight of the edge e : U → V in the retimed graph is computed from the weight of the edge in the original graph using

$$\omega_r(e) = \omega(e) + r(V) - r(U)\ r(V), r(U) \ldots\ldots\ldots\ldots(1)$$

Where Z is the set of Integers.

The retiming values r(1) = 0, r(2) = 1, r(3) = 0 and r(4) = 0 translates the DFG in Fig. 2.1(a) to the retimed DFG in Fig. 2.1(b). Retiming does not alter the architecture of the design, hence the incidence and loop matrices for the design remain for the original and retimed DFG. However, the weights on the edges change and it is the weight vector which is transformed during retiming.

A retiming solution is feasible if $\omega_r(e) \geq 0$ holds for all edges. It can be proved that retiming does not alter the total number of delays in a loop of the DFG . This would mean that for a given loop in the DFG the sum of delays on the edges of the loop will remain unchanged after retiming.

III. LITERATURE REVIEW

SR. NO.	TITLE	AUTHOR	YEAR	METHODOLODY
1	On Efficient Retiming of Fixed-Point Circuits,	P. K. Meher,	2016	Connected component timing model to obtain adequately precise estimates of propagation delays across different combinational paths in a DFG easily
2	Low-complexity CRC-aided early stopping unit for parallel turbo decoder,	H. Kim, Y. Lee and J. H. Kim,	2015	CRC architecture is implemented in 65 nm CMOS process for radix-2^2 and radix-2^4 parallel turbo decoders based on LTE-Advanced.
3	Fine-Grained Critical Path Analysis and Optimization for Area-Time Efficient Realization of Multiple Constant Multiplications,	X. Lou, Y. J. Yu and P. K. Meher,	2015	The DOCO is applied to each searched additional fundamental set to optimize the configuration of the corresponding shift-add network.
4	Critical-Path Analysis and Low-Complexity Implementation of the LMS Adaptive Algorithm,	P. K. Meher and S. Y. Park,	2014	The critical path of the least-mean-square (LMS) adaptive filter for deriving its architectures for high-speed and low-complexity implementation
5	Efficient Retiming of Multirate DSP Algorithms,	X. Y. Zhu, T. Basten, M. Geilen and S. Stuijk,	2012	A lower iteration period implies a faster execution of a DSP algorithm with synchronous dataflow graphs (SDFGs)
6	Static Rate-Optimal Scheduling of Multirate DSP Algorithms via Retiming and Unfolding	X. Y. Zhu, M. Geilen, T. Basten and S. Stuijk,	2012	An exact method and a heuristic method for static rate-optimal multiprocessor scheduling of real-time multi rate DSP
7	A Fast Retiming Algorithm Integrated with Rewiring for Flip-Flop Reductions,	Y. Diao and Y. Wu,	2011	A fast retiming algorithm which avoids solving MILP, and with both gate and interconnect delay formulated together.

P. K. Meher,[1] Retiming of digital circuits is conventionally based on the estimates of propagation delays across different paths in the data-flow graphs (DFGs) obtained by discrete component timing model, which implicitly assumes that operation of a node can begin only after the completion of the operation(s) of its preceding node(s) to obey the data dependence requirement. Such a discrete component timing model very often gives much higher estimates of the propagation delays than the actuals particularly when the computations in the DFG nodes correspond to fixed-point arithmetic operations like additions and multiplications. On the other hand, very often it is imperative to deal with the DFGs of such higher granularity at the architecture-level abstraction of digital system design for mapping an algorithm to the desired architecture, where the overestimation of propagation delay leads to unwanted pipelining and undesirable increase in pipeline overheads. In this research work, propose the connected component timing model to obtain adequately precise estimates of propagation delays across different combinational paths in a DFG easily, for efficient cutset-retiming in order to reduce the

critical path substantially without significant increase in register-complexity and latency. Apart from that, propose novel node-splitting and node-merging techniques that can be used in combination with the existing retiming methods to achieve reduction of critical path to a fraction that of the original DFG with a small increase in overall register complexity.

H. Kim, Y. Lee and J. H. Kim, [2] A low-complexity distributed cyclic redundancy check (CRC) architecture for the CRC-aided early stopping unit is proposed. In the previous distributed CRC unit, the general high-order Galois field (GF) multiplier occupies almost the area of the CRC unit and requires high-hardware cost and long critical path-delay. Accordingly, a computation algorithm based on GF arithmetic is analysed and an optimal CRC unit with the small order of the GF multiplier and newly designed linear feedback shift register is proposed. The proposed CRC architecture is implemented in 65 nm CMOS process for radix-22 and radix-24 parallel turbo decoders based on LTE-Advanced. In the radix-22 system, reductions of about 57.1% of gate count, 31.7% of critical path-delay and 44.1% of power consumption are achieved compared with the previous work.

X. Lou, Y. J. Yu and P. K. Meher, [3] In this research work, critical path of multiple constant multiplication (MCM) block is analyzed precisely and optimized for high-speed and low-complexity implementation. A delay model based on signal propagation path is proposed for more precise estimation of critical path delay of MCM blocks than the conventional adder depth and the number of cascaded full adders. A dual objective configuration optimization (DOCO) algorithm is developed to optimize the shift-add network configuration to derive high-speed and low-complexity implementation of the MCM block for a given fundamental set along with a corresponding additional fundamental set. A genetic algorithm (GA)-based technique is further proposed to search for optimum additional fundamentals. In the evolution process of GA, the DOCO is applied to each searched additional fundamental set to optimize the configuration of the corresponding shift-add network. Experimental results show that the proposed GA-based technique reduces the critical path delay, area, power consumption, area delay product and power delay product by 32.8%, 4.2%, 5.8%, 38.3%, and 41.0%, respectively, over other existing optimization methods.

P. K. Meher and S. Y. Park,[4] This research work presents a precise analysis of the critical path of the least-mean-square (LMS) adaptive filter for deriving its architectures for high-speed and low-complexity implementation. It is shown that the direct-form LMS adaptive filter has nearly the same critical path as its transpose-form counterpart, but provides much faster convergence and lower register complexity. From the critical-path evaluation, it is further shown that no pipelining is required for implementing a direct-form LMS adaptive filter for most practical cases, and can be realized with a very small adaptation delay in cases where a very high sampling rate is required. Based on these findings, this research work proposes three structures of the LMS adaptive filter: (i) Design 1 having no adaptation delays, (ii) Design 2 with only one adaptation delay, and (iii) Design 3 with two adaptation delays. Design 1 involves the minimum area and the minimum energy per sample (EPS). The best of existing direct-form structures requires 80.4% more area and 41.9% more EPS compared to Design 1. Designs 2 and 3 involve slightly more EPS than the Design 1 but offer nearly twice and thrice the MUF at a cost of 55.0% and 60.6% more area, respectively.

X. Y. Zhu, T. Basten, M. Geilen and S. Stuijk,[5] Multirate digital signal processing (DSP) algorithms are often modeled with synchronous dataflow graphs (SDFGs). A lower iteration period implies a faster execution of a DSP algorithm. Retiming is a simple but efficient graph transformation technique for performance optimization, which can decrease the iteration period without affecting functionality. In this research work, deal with two problems: feasible retiming-retiming a SDFG to meet a given iteration period constraint, and optimal retiming-retiming a SDFG to achieve the smallest iteration period. present a novel algorithm for feasible retiming and based on that one, a new algorithm for optimal retiming, and prove their correctness. Both methods work directly on SDFGs, without explicitly converting them to their equivalent homogeneous SDFGs. Experimental results show that our methods give a significant improvement compared to the earlier methods.

X. Y. Zhu, M. Geilen, T. Basten and S. Stuijk,[6] This research work presents an exact method and a heuristic method for static rate-optimal multiprocessor scheduling of real-time multi rate DSP algorithms represented by synchronous data flow graphs (SDFGs). Through exploring the state-space generated by a self-timed execution (STE) of an SDFG, a static rate-optimal schedule via explicit retiming and implicit unfolding can be found by our exact method. By constraining the number of concurrent firings of actors of an STE, the number of processors used in a schedule can be limited. Using this, present a heuristic method for processor-constrained rate-optimal scheduling of SDFGs. Both methods do not explicitly convert an SDFG to its equivalent homogenous SDFG. Our experimental results show that the exact method gives a significant improvement compared to the existing methods, our heuristic method further reduces the number of processors used.

Y. Diao and Y. Wu,[7] Traditional retiming processes are mostly MILP based with the physical interconnect information less correctly reflected, thus could be very CPU intensive and incorrect on the final clock period estimations. Moreover, the number of flip-flops tends to be undesirably increased after the retiming process, which can cause a significant area/power penalty on the retimed circuit. To overcome these major drawbacks of the conventional retiming technique, first propose a fast retiming algorithm which avoids solving MILP, and with both gate and interconnect delay formulated together. For a more accurate delay estimation, all interconnect delays are formulated and calculated based on real placements. Additionally, integrate it with a specific rewiring algorithm to cut down the number of flip-flops (FFs) without sacrificing retimed clock periods. Experimental results show that our pure retiming algorithm can achieve an average of 5.75% optimization on the clock period in a very fast speed. With a rewiring algorithm targeting for FF reduction applied, a FF reduction of up to 31.2% (11.3% on average) can be obtained without the compromise on the retimed clock period compared to the pure retimed results.

IV. PROBLEM STATEMENT

Optimal decision making has many practical advantages such as allowing for a systematic design of the decision maker or improving the quality of the decisions taken in the presence of constraints. However, the need to solve an optimization problem at every decision instant, typically via numerical iterative algorithms, imposes a very large computational demand on the device implementing the decision maker. Consequently, so far, optimization-based decision making has only been widely adopted in situations that require making decisions only once during the design phase of a system, or in systems that, while requiring repeated decisions, can afford long computing times or powerful machines.

Implementation of repeated optimal decisions on systems with resource constraints remains challenging. Resource constraints can refer to:

1) Time - the time allowed for computing the solution of the optimization problem is strictly limited,
2) The computational platform - the power consumption, cost, size, memory available, or the computational power are restricted

V. CONCLUSION

This work presents an extensive survey on efficient retiming of fixed-point circuits. There are various retiming algorithms have been proposed for efficient retiming and significant amount of research has been done on retiming. An introduction to retiming and its application merits and demerits have been presented in this work. Also a brief survey of literature has been done. Based on the literature survey problem has been formulated for existing retiming algorithms and recent approaches. The major approaches in retiming circuits are Edge-triggered Circuits, Timing Models, Level-Clocked Circuits, Retiming with Equivalent Initial States. Retiming can alter the amount of switching that takes place in a circuit, and can, therefore, affect the power consumption of a circuit and can be used both to improve testability of a circuit, and as an aid to automatic test generation. Retiming relocates the memory elements in a circuit without changing its behavior.

REFERENCES

[1] P. K. Meher, "On Efficient Retiming of Fixed-Point Circuits," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 4, pp. 1257-1265, April 2016.

[2] H. Kim, Y. Lee and J. H. Kim, "Low-complexity CRC-aided early stopping unit for parallel turbo decoder," in Electronics Letters, vol. 51, no. 21, pp. 1660-1662, 10 8 2015.

[3] X. Lou, Y. J. Yu and P. K. Meher, "Fine-Grained Critical Path Analysis and Optimization for Area-Time Efficient Realization of Multiple Constant Multiplications," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 62, no. 3, pp. 863-872, March 2015.

[4] P. K. Meher and S. Y. Park, “Critical-path analysis and low-complexity implementation of the LMS adaptive algorithm,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 61, no. 3, pp. 778-788, Mar. 2014.

[5] X. Y. Zhu, T. Basten, M. Geilen and S. Stuijk, "Efficient Retiming of Multirate DSP Algorithms," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 31, no. 6, pp. 831-844, June 2012.

[6] X. Y. Zhu, M. Geilen, T. Basten and S. Stuijk, "Static Rate-Optimal Scheduling of Multirate DSP Algorithms via Retiming and Unfolding," 2012 IEEE 18th Real Time and Embedded Technology and Applications Symposium, Beijing, 2012, pp. 109-118.

[7] Y. Diao and Y. Wu, "A Fast Retiming Algorithm Integrated with Rewiring for Flip-Flop Reductions," 2011 12th International Conference on Computer-Aided Design and Computer Graphics, Jinan, 2011, pp. 471-477.

[8] M.R.C.M. Berkelaar. LP.SOLVE User’s Manual. Eindhoven University of Technology, Eindhoven, The Netherlands, 1992.

[9] F. Brglez, D.Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In Proceedings of the IEEE International Symposium on Circuits and Systems, pages 1929-1934, 1989.

[10] T. M. Burks and K. A. Sakallah. Optimization of critical paths in circuits with level-sensitive latches. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, pages 468-473, 1994.

[11] T. M. Burks, K. A. Sakallah, and T. N. Mudge. Critical paths in circuits with level- sensitive latches. IEEE Transactions on VLSI Systems, 3(2):273—291, June 1995.