On the Optimization Power of Redundancy Addition and Removal Techniques for Sequential Circuits

Enrique San Millán, Luis Entrena, José Alberto Espejo
Electrical, Electronic and Automation Engineering Department
University Carlos III of Madrid
{quique, entrena, ppespejo}@ing.uc3m.es

ABSTRACT

This paper attempts to determine the capabilities of existing Redundancy Addition and Removal (SRAR) techniques for logic optimization of sequential circuits. To this purpose, we compare this method with the Retiming and Resynthesis (RaR) techniques. For the RaR case the set of possible transformations has been established by relating them to STG transformations by other authors. Following these works, we first formally demonstrate that logic transformations provided by RaR are covered by SRAR as well. Then we also show that SRAR is able to identify transformations that cannot be found by RaR. This way we prove the higher potential of the Sequential Redundancy Addition and Removal over the Retiming and Resynthesis techniques.

1. INTRODUCTION

Logic optimization for synchronous sequential circuits is still an open and challenging problem. Given an initial circuit description at the gate level, sequential logic optimization aims at computing an equivalent circuit with a smaller area occupation usually estimated through the gate, connection or literal counts.

The two main approaches to sequential logic optimization are Retiming and Resynthesis (RaR) [MSBS91] and Sequential Redundancy Addition and Removal (SRAR) [EnCh95]. The Retiming technique consists in moving the flip-flops across combinational gates, merging combinational blocks and optimizing the resulting logic with combinational techniques in a Resynthesis step. This method has become quite attractive, despite its limitations.

The possible movements of flip-flops across combinational gates can be built from a sequence of four primitive retiming operations, as stated in the following lemma [RSSB98]:

Lemma: A general retiming operation can be constructed as the sequence of retiming moves across primitive transformations i), ii), iii) and iv) shown in the figure 1.1.

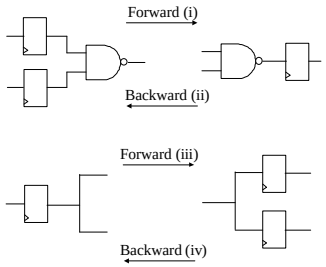

Figure 1.1: Primitive retiming operations

The method of Retiming and Resynthesis has been widely studied, and the optimization capability of these methods have been formally established by several authors [MSBS91][ZSA98][Ran97][RSSB98].

The approach to establish the capabilities of the method has been characterized by relating them to the STG transformations in the following theorems [RSBS91][RSSB98]:

Theorem (Malik): Given a machine implementation M1, corresponding to a state transition graph G, with a state assignment S1, it is always possible to derive a machine M2 corresponding to the same state transition graph G, and a state assignment S2 by applying only a series of Resynthesis and Retiming operations on M1

Theorem: Let M1 be an implementation corresponding to state assignment S1 and STG G1 and M2 be an implementation corresponding to state assignment S2 and STG G2. Then M2 can be obtained from M1 using only a sequence of Retiming and Resynthesis operations if and only if G1 and G2 are 1-step equivalent.

The other method, Redundancy Addition and Removal has been shown to be a powerful logic optimization method by several authors [EnCh95], [ChMa94], [KuMe94], [EESO01]. This technique has demonstrated to produce excellent results for combinational circuits, being the major advantages the low memory usage and the short run times. Sequential logic optimization is also possible using this technique by considering sequential redundancies [EnCh95].

In this work we compare the sequential transformations that can be obtained by RaR and by SRAR, working towards a formalization of the set of transformations that can be obtained with the latter technique. In particular, we demonstrate that the SRAR technique covers all the possible optimizations that can be provided by the RaR technique. Then we will show that SRAR is able to optimize some typical circuit cases that cannot be optimized by Retiming and Resynthesis. This way, we demonstrate the higher potential of the Sequential Redundancy Addition and Removal technique.

2. OPTIMIZATION POWER OF SEQUENTIAL REDUNDANCY ADDITION AND REMOVAL

2.1 Possible Retiming transformations with Redundancy Addition and Removal

Although Redundancy Addition and Removal techniques have been shown to be efficient optimization techniques, the power of optimization of these methods has not been formalized. In this section we will show some of the capabilities of these methods, first comparing them with the Retiming methods and then showing some additional features.

Theorem: All possible Retiming transformations in a circuit can be obtained by Redundancy Addition and Removal Transformations.

Proof: As all possible Retiming transformations are compositions of four primitive operations, we just need to proof that each one of these primitives can be performed by a Redundancy Addition and Removal Transformation. We consider the four cases i), ii), iii) and iv) shown in figure 1.1:

i) Moving forward two registers through a nand gate can be accomplished in two steps: moving first the inverter, and then the and gate. So we consider both cases:

- Moving forward an and gate:

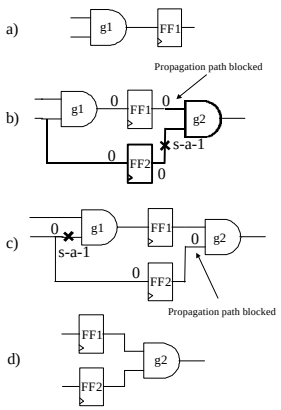

Figure 2.2: Moving forward an and gate.

Consider the circuit in figure 2.2 a). We need to move forward g1 through FF1. First we add g2 and FF2 as shown in figure 2.2 b) because the connection between g2 and FF2 is redundant. To show this redundancy, consider the s-a-1 fault shown in figure 2.2 b). To justify the fault, FF2 needs a mandatory assignment of 0, and by implication, g1 and FF1 have assignments of 0. FF1=0 blocks the fault, and thus connection FF2 to g2 is redundant.

Now, by adding this redundancy to the circuit, a new redundancy has appeared as shown in figure 2.2 c). S-a-1 fault shown in that figure produces a mandatory assignment of 0 in g2, which blocks the fault propagation. So the second input to g1 is redundant, and by eliminating this redundancy we get 2.2 d), reaching the transformed circuit we were looking for.

- Moving forward an inverter:

Consider the circuit in figure 2.3a). The addition shown in 2.3b) and removal shown in 2.3c) leads to 2.3d)

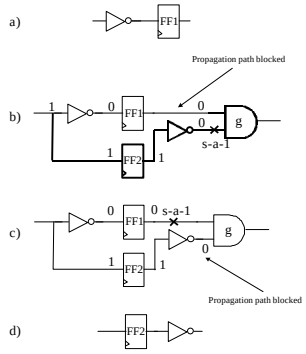

Figure 2.3: Moving forward an inverter

ii) *Moving backward a register through a nand gate:*

It can be accomplished as in i) in two steps: moving first the and gate and then moving the inverter.

- Moving backward an and gate:

Consider the circuit in figure 2.4a). The addition shown in 2.4b) and removal shown in 2.4c) leads to 2.4d)

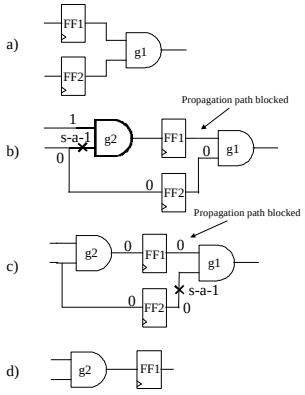

Figure 2.4. Moving backward an and gate

- Moving backward an inverter:

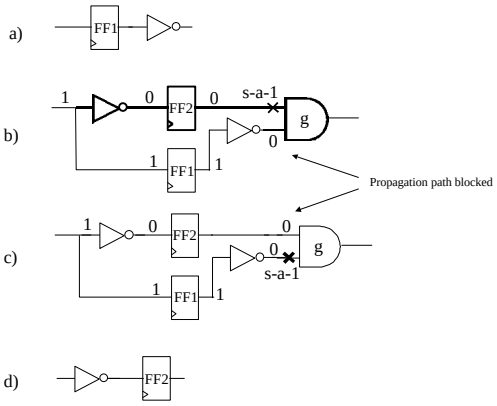

Figure 2.5. Moving backward an inverter

Consider the circuit in figure 2.5a). The addition shown in 2.5b) and removal shown in 2.5c) leads to 2.5d)

iii) Moving forward a register in a multiple fanout point:

Consider the circuit in figure 2.6a). The addition shown in 2.6b) and removal shown in 2.6c) leads to 2.6d)

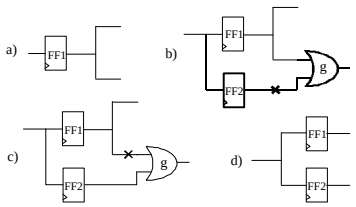

Figure 2.6. Moving forward a register in a multiple fanout point

iv) Moving backward a register to a multiple fanout point:

Consider the circuit in figure 2.7a). The addition shown in 2.7b) and removal shown in 2.7c) leads to 2.7d)

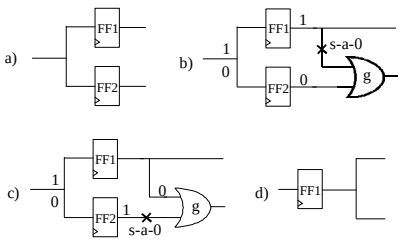

Figure 2.7

Having proved that all the primitive Retiming transformations can be achieved by Redundancy Addition and Removal transformations, the proof is finished. ■

This theorem shows that the powerful of Redundancy Addition and Removal techniques is at least the same as the one stated for Retiming methods.

Note that for the proof of the theorem only a subset of all the possible redundancy additions is used. Addition of multiple wires/gates at the same time is not needed for the proof.

As we have characterization of the possible Retiming transformations by relating them to the STG transformations, we apply those results to Redundancy Addition and Removal methods as follows:

Corollary: (encoding power of Redundancy Addition and Removal). Given a machine implementation M1, corresponding to a state transition graph G, with a state assignment S1, it is always possible to derive a machine M2 corresponding to the same state transition graph G, and state assignment S2 by applying only a series of Redundancy Addition and Removal operations on M1.

Corollary: Let M1 be an implementation corresponding to state assignment S1 and STG G1 and M2 be an implementation corresponding to state assignment S2 and STG G2 such that G2 is 1-step equivalent to G1. Then M2 can be obtained from M1 using only a sequence of Redundancy Addition and Removal operations.

2.2 Redundancy Addition and Removal Transformations which are not possible with Retiming and Resynthesis

Now we have established some of the capabilities of the Redundancy Addition and Removal methods, but now we will show that, opposed to the Retiming methods, these are not the only possible STG transformations that these techniques provide.

We will show examples where optimization by Retiming and Resynthesis is not possible, but optimization by Redundancy Addition and Removal is.

Example 1: In the example shown in figure 2.8 is not possible to reach circuit 2.8b) from circuit 2.8a) by a sequence of Retiming and Resynthesis transformations [Ran97].

However, Redundancy Addition of a gate and Removal of a wire makes the optimization possible. The addition of the redundant or gate shown in figure 2.8c) makes the wire shown in figure 2.8d) redundant, and the removal of this redundant wire leads to circuit 2.8b).

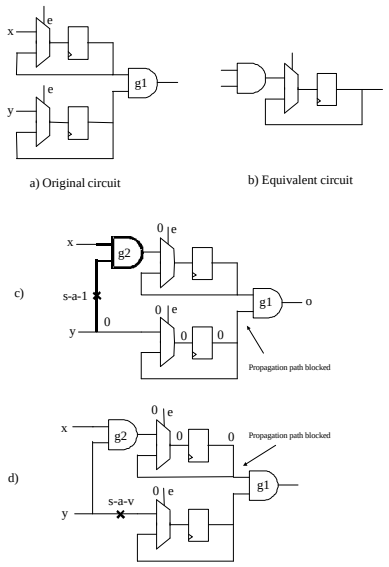

Figure 2.8. Example 1

Example 2: In the example of figure 2.9 is not possible to reach circuit 2.9b) from 2.9a) by a sequence of Retiming and Resynthesis transformations [RSSB98].

As in the previous example, there is a transformation of Addition and Removal that can transform one circuit into the other. The addition of the redundant or gate shown in figure 2.9c), makes redundant the wire shown in figure 2.9d). Removal of that redundant gate leads to circuit 2.9b)

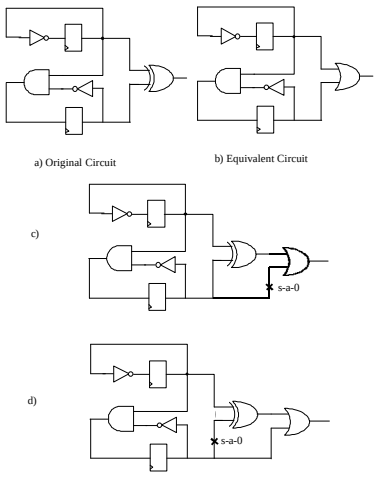

Figure 2.9. Example 2

3. CONCLUSIONS AND FUTURE WORK

Logic optimization for synchronous sequential circuits is still an open issue. The two main approaches to sequential logic optimization are RaR and SRAR. In this work we have compared the sequential transformations that can be obtained with both methods.

We have formally demonstrated that all the possible transformations that can be obtained in a circuit by a sequence of RaR operations can also be obtained by a sequence of SRAR transformations.

We have also shown that SRAR is able to perform transformations that cannot be found by RaR. This way we have proved the SRAR techniques are more complete than the RaR methods.

4. REFERENCES

[ChMa94] S. C. Chang, M. Marek-Sadowska. “Perturb and Simplify: Multi-level Boolean Network Optimizer”. Proc. ICCAD-94, p. 2-5. November, 1994

[EnCh95] L.A. Entrena, K.-T. Cheng. “Combinational and sequential logic optimization by redundancy addition and removal”. IEEE Trans. on CAD, Vol. 14, No. 7, July 1995, p. 909-916

[GlCh95] U. Gläser, K.-T. Cheng. “Logic Optimization by an Improved Sequential Redundancy Addition and Removal Technique”. Proc. ASP-DAC. September, 1995

[KuMe94] W. Kunz, P. R. Menon. “Multi-level Logic Optimization by Implication Analysis”. Proc. ICCAD-94, pp. 6-13. November, 1994

[MSBS91] S. Malik, E. M. Sentovich, R. Brayton, A. Sangiovanni-Vincentelli. “Retiming and Resynthesis: Optimizing Sequential Circuits Using Combinational Techniques”. IEEE Transactions on CAD of Integrated Circuits and Systems, vol. 10, p. 74-84. January 1991

[EESO01] J. A. Espejo, L. Entrena, E. San Millán, E. Olías. “Generalized Reasoning Scheme for Redundancy Addition and Removal Logic Optimization”. Proc. DATE’01, p. 391-395. March 2001.

[SEEO99] E. San Millán, L. Entrena, J.A. Espejo, Silvia Chiusano, Fulvio Corno. “ Integrating symbolic Techniques in ATPG-Based Sequential Logic Optimization”. Proc. DATE’99, p. 516-523 March 1999.

[RSSB98] R.K. Ranjan, V. Singhal, F. Somenzi, R.K. Brayton. “On the optimization Power of Retiming and Resynthesis Transformations”. Proc. ICCAD’98, p 402-407, November 1998.

[Ran97] R.K.Ranjan. “Design and Implementation Verification of Finite State Systems”. PhD thesis. Electronics Research Laboratory. University of California. Berkeley. CA 94720. 1997. Memorandum No. UCB/ERL M97/99.

[ZSA98] H.Zhou, V. Singhal, A. Aziz. “How Powerful is Retiming?”. Proc. IEEE/ACM intl. Workshop on Logic Synthesis, p 111-125, May 1998.