Using Essential Inverters for Interconnect Delay Reduction

Lung-Jen Lee and Rung-Bin Lin
Computer Science and Engineering
Yuan Ze University, Taiwan
lauren@vlsi.cse.yzu.edu.tw, csrlin@cs.yzu.edu.tw

ABSTRACT

With the fast advance of VLSI process technology, interconnect delay increasingly dominates the circuit performance. Buffer insertion plays a crucial role in dealing with this problem. However, excessive insertion might cause additional problems and counteract its advantages. In this paper, we propose a gate replacement method to extract essential inverters from positive unate gates. The basic idea is to use inverters originally embedded in a design rather than externally added buffers to drive long interconnects. Our experiments show on average up to 27 % reduction in buffer usage together with 4.6% reduction in clock period. The total slack of the first 100 longest paths is improved by 67.1%. The total negative slack is improved by 53.9%. All these are achieved at the expense of on average 3.1% increase in cell area for the large benchmark circuits.

1. Introduction

As the process technology advances into the deep sub-micron era, interconnect delay plays an increasingly critical role in affecting circuit performance and signal integrity [1]. Many approaches are proposed to deal with the interconnect delay problem. Most of these approaches either modify the logic or change the physical aspects of a circuit [2]. Physical techniques include gate sizing, buffer insertion [3], gate relocation [4], etc. Logical solutions include gate replication [5], rewiring [6,7], restructuring [8, 9,10,11,12], etc. Among them, buffer insertion is a very effective and popular technique in reducing the interconnect delay [1]. An inserted buffer strengthens a signal to drive many sinks or long wires. However, an industry study [13] predicts that buffers will take about 15% of the cells on a chip at the 65*nm* technology node. Assuming the continuation of the current scaling trend and design styles, it is estimated that buffers will take about 70% of the cells at the 32nm technology node [13]. The huge number of buffers may affect various aspects of circuit designs and performances including timing [14], power dissipation [15], signal integrity [16], placement, and routing congestion [17,18]. Therefore, buffer insertion needs to be conducted in a more efficient manner to reduce the number of buffers used for interconnect delay optimization.

In this paper, we propose a gate replacing method which replaces AND (OR) gates with NAND (NOR) and NOT gates to improve timing performance and reduce buffer usages at the same time. Our basic idea is to use inverters originally embedded in a design rather than externally added buffers to drive long interconnects. As shown in Fig. 1, the AND gate and a buffer on the top is replaced by a NAND gate and an inverter. The so-obtained circuit not only uses one fewer buffer but also achieves better timing because the delay of a NAND gate is smaller than that of an AND gate and an inverter's delay is smaller than a buffer's delay. Besides, the circuit also consumes less power. Table 1 shows the total gate counts and the percentage of AND (OR) gates in each benchmark circuit. As one can see, there is a considerable percentage of AND (OR) gates in each benchmark circuit. AND gates or OR gates are a kind of positive unate gates. The output transition direction of a positive unate gate is the same as its input transition direction. On the contrary, NAND gates and NOR gates are called negative unate gates. We call the inverter embedded in a positive unate gate an essential inverter. We extract essential inverters after a design is done with logic synthesis. Once we complete the replacement of positive unate gates, we carry out circuit placement, routing, and post-layout timing analysis. All these are done with a help from a commercial tool suite. Compared to the results attained for some large benchmark circuits by a commercial tool with buffer insertion, our approach on average achieves up to 27% reduction in buffer usage together with 4.6% reduction in clock period. The total slack of the first 100 longest paths is improved by 67.1%. The total negative slack is improved by 53.9% and the total wire length is on average reduced by 0.5%. All these are achieved at the expense of up to 3.1% increases in cell area.

The rest of this paper is organized as follows. In Section 2 we discuss the concept of using essential inverters to reduce interconnect delay and buffer usage. We also present our gate replacement method. Section 3 presents some experimental results. The last section draws a conclusion and discusses possible future work.

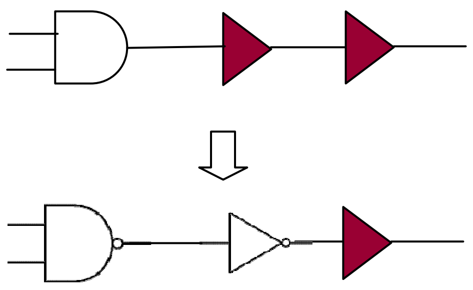

Fig. 1. Splitting an AND gate into a NAND and an essential inverter.

Table 1. Gate counts and AND/OR gate percentages.

Benchmark	circuit	cell counts	AND/OR(%)
ITC99'	b18	36430	11.4%
	b19	73500	11.4%
ISCAS. 89'	S35932	6156	11.8%
	S38417	4850	9.2%
	S38584	5669	23.1%

2. Method

Our method is enabled by having a gate replacement program to replace a positive unate gate by a negative unate gate plus an essential inverter. For ease of presentation, the circuit prior to any transformation is called original circuit. The one generated by the gate replacement program is called a transformed circuit. Some other circuit transformations of the same logic function are given in Fig. 2. The gate replacement program can be run prior to placement or subsequent to placement. It can be designed to replace all positive unate gates in a circuit or selectively replace some positive unate gates on critical paths. In this section we will first discuss some problems in general related to gate replacement. We then specifically discuss our gate replacement approach.

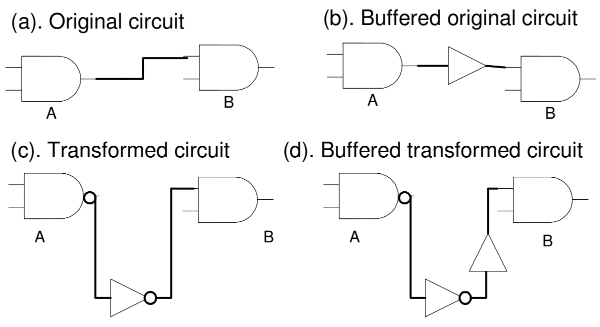
(a). Original circuit
A
B
(b). Buffered original circuit
A
B
(c). Transformed circuit
A
B
(d). Buffered transformed circuit
A
B

Fig. 2. Different circuit transformations of the same logic function.

2.1. Problems Related to Gate Replacement

Although essential inverters play a similar role as buffers do in improving timing performance, placement of the essential inverters on signal propagation paths is more restricted. As shown in Fig. 3, we allow any of the four situations to happen to a buffer or buffers. However, as shown in Fig. 4, we allow either only case (a) or (d) to happen to an inverter or inverters. The circuit in Fig. 3(a) uses four more transistors than the circuit in Fig. 4(a) does and two more transistors than the circuit in Fig. 4(d) does. Although the circuit in Fig. 4(d) has two fewer transistors, its total cell area is not necessarily smaller than that of the circuit in Fig. 3(a). The reason for this is simple. The total area of two inverter cells is larger than a buffer cell and the total area of a NAND cell and an inverter cell is larger than the area of an AND cell if the sizes of the transistors in the former are the same as those of the corresponding transistors in the latter. This is the reason why the total cell area of a transformed circuit is normally larger than that of the original circuit. However, when compared to the total cell area of a buffered original circuit, the total cell area of a transformed circuit can be smaller than that of the buffered original circuit. In any situation, a transformed circuit has a smaller total transistor count, which means a smaller power consumption

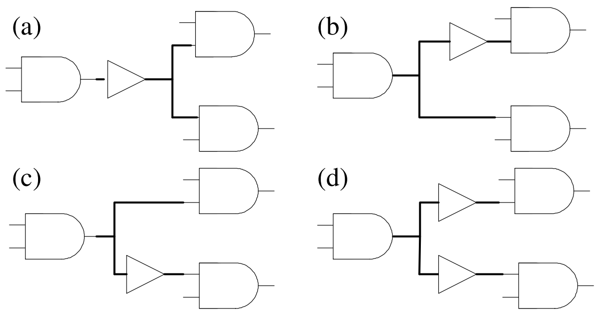

Fig.3. Placement of buffers.

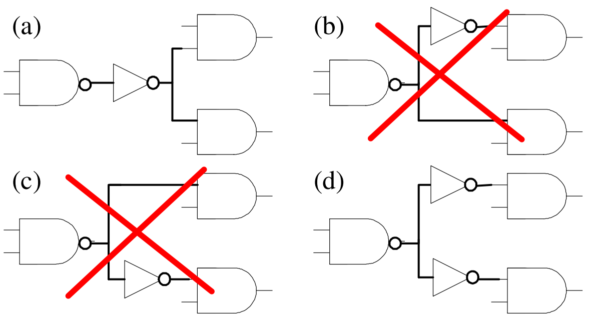

Fig. 4. Placement of essential inverters.

Besides being using more cell area when compared to an original circuit, a transformed circuit also has a larger cell count and thus has more nets. This may make a physical design tool more difficult to obtain a good solution. It can also create a problem of routing detour in a transformed circuit if gate replacement is done before placement. As shown in Fig. 2, due to a non-optimal placement of the inverter in the transformed circuit given in Fig. 2(c), the path length from point A to point B is in fact increased and the propagation delay between these two points may be thus increased.

Other basic problem related to replacing a positive unate gate by a negative unate gate and an essential inverter is to determine the driving strengths of the essential inverter and the negative unate gate. Basically, the wire load and fanout driven by these gates should be taken into account. This can be more accurately done if gate replacement is done after placement.

Another problem is that which of the gate replacement ways presented in Fig. 4(a) and 4(d) should be used. It is clear that the way given in Fig. 4(a) is more preferable. Yet another problem that relates to post-placement gate replacement is to find an empty space to hold the essential inverter so that the placement can achieve better timing performance. This is similar to the post-placement buffer insertion problem.

Some of the above problems may offset the figure of merit of using essential inverters for interconnect delay optimization, while the other may bring challenges to the methods designed to solve the underlying problems.

2.2. Our Gate Replacement Method

For the time being our gate replacement program is designed to replace all positive unate gates prior to placement. It is the most intuitive and simplest approach as shown in Fig. 5. Despite its simplicity, it is difficult to determine the driving strengths of the essential inverter and the negative unate gate. To respect the original decision made by a logic synthesis tool, the driving strength of the essential inverter is set equal to that of the positive unate gate which is being replaced. The driving strength of the negative unate gate is set equal to either 1X, 2X, or *m*X where *m* is the driving strength of the positive unate gate. Therefore, we denote the three ways of setting the driving strengths of the negative unate gate and the essential inverter by 1:*m*, 2:*m* and *m*:*m*, respectively. In each way, the first figure represents the driving strength of the negative unate gate and the second figure represents that of the essential inverter. For the example shown below, a positive unate gate U7363 with driving strength 4X is replaced by a 4X inverter and a 1X negative unate gate.

<Example >
Original gate level circuit:
AND2X<u>4</u> U7363 (.A(n8356), .B(n7124), .Y(n6240));
Transformed gate level circuit:
NAND2X<u>1</u> U7363 (.A(n8356), .B(n7124), .Y(n9001));
INVX<u>4</u> U7601 (.A(n9001), .Y(n6240));

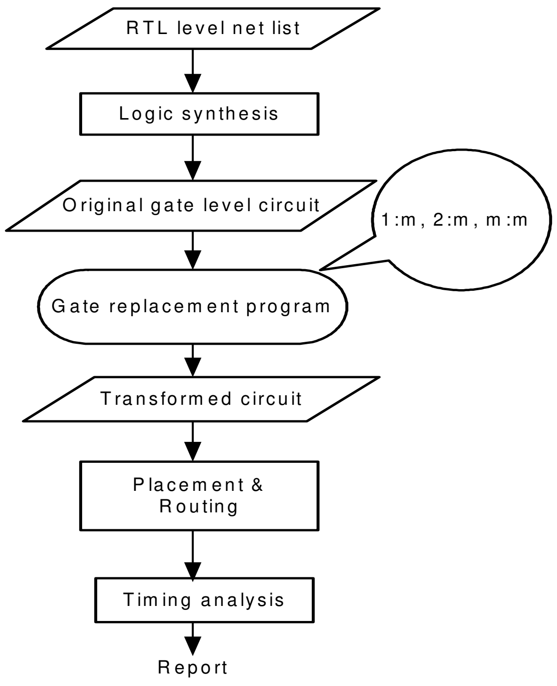

Fig. 5. A design flow using our gate replacement method.

3. Experimental Results

The test circuits used in our experiments are shown in Table 1. Two kinds of experiments are performed. The first kind is performed to see whether a transformed circuit will have better timing behavior than the original circuit. This kind of experiments does not insert buffers into the original circuit. The other kind is performed to see whether a buffered transformed circuit will have better timing behavior than the buffered original circuit and see whether the use of buffers can be reduced. Commercial tools are used to perform logic synthesis, placement, routing, and timing analysis. A TSMC 0.18 um standard cell library is used.

Table 2 shows the results obtained by the first kind of experiments, where the column denoted by *org* gives the data for the original circuit and the column denoted by *NULL* gives the data for the circuit without using any positive unate gates (forcing a logic synthesis tool not to use any AND and OR gates). The columns denoted by 1:*m*, 2:*m*, and *m*:*m* gives the data for the transformed circuits with different driving strength setting strategies, respectively. For larger circuits b18 and b19, mechanism 2:*m* achieves on average 3.8% improvement in clock period, 134.5% in the total slack of the first 100 worst case paths, and 67.1% in total negative slack at the expense of on average 2.1% increase in total wire length and 5% in total cell area. For smaller circuits s35932, s38417 and s38584, no consistent improvements are observed. Some of which even have a larger clock period. This is not surprising if one refers to the arguments given in Section 2.1.

Table 3 presents the results for the second kind of experiments. For larger circuits b18 and b19, mechanism *m*:*m* achieves on average 4.6% improvement in clock period, 67.1% in the total slack of the first 100 worst case paths, 53.9% in total negative slack, and 27.0% in buffer reduction at the expense of on average 3.1% increase in total cell area. For smaller circuits, no consistent improvements are observed.

We must admit that the gate replacement method currently implemented in this paper is very primitive. However, with such a primitive method, improvement in clock period, negative slacks, and buffer usage has been constantly observed. We predict that a more sophisticated post placement gate replacement method based on critical path information would be more fruitful.

4. Conclusions

In this paper we have discussed the concept of using essential inverters to improve interconnect delay and reduce buffer usage. We also depict the problems that need to be addressed for any gate replacement method. Although our gate replacement is simple yet effective, a more sophisticated gate replacement method performed after placement based on critical path information would even bring better results. This is the work yet to be done in the future.

Table 2. Result comparison of transformed circuits and original circuit.

	clock period					100-Worst-Paths' total slack					total negative slack					total wire length					total cell area				
	ns	% improved				ns	% improved				ns	% improved				m	% increased				10^5	% increased			
	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m
b18	22.3	-4.9	0.9	3.6	4.5	-41	-288.9	131.6	201.4	226.9	-64	-157	38.0	67.4	75.2	2.62	-0.7	1.2	1.3	2.9	9.32	6.7	2.0	5.1	3.3
b19	22.6	-2.7	1.3	4.0	3.5	-143	-68.5	28.4	67.6	71.7	-153	-92.1	31.1	66.8	67.7	5.53	-3.9	0.9	2.9	-2.5	1.87	6.7	1.9	4.9	3.0
s35932	7.1	-20.1	-6.6	-10	-14.4	133	-132	-44.7	-62.6	-72.9	0	-80.3*	0.0	0.0	-0.2*	0.36	2.7	2.5	1.9	4.6	1.72	-1.8	2.8	5.7	5.7
s38417	7.4	13.1	4.1	-4.2	-1.2	122	-76.2	-17.2	1.6	0.4	0	0.0	0.0	0.0	0.0	0.27	-1.8	1.5	3	1.8	1.56	-1.5	1.8	4.0	3.6
s38584	8.2	0.5	11.0	-43.9	25.3	125	132.8	55.2	20	98.2	0	0.0	0.0	0.0	0.0	0.39	0.15	2.1	5.3	3.5	1.52	-1.7	5.3	12.0	10.9

* Note that the numbers with "*" in the column of *total negative slack* give the total negative slack values.

Table3. Result comparison of buffered transformed circuits and buffered original circuit.

	clock period					buffer usage					100-Worst-Paths' total slack					total negative slack					total wire length					total cell area				
	ns	% improved				#	% reduction				ns	% improved				ns	% improved				m	% increased				10^5	% increased			
	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m	org	NULL	1:m	2:m	m:m
b18	23	-10.4	-1.1	3	4.7	95	43.2	8.4	23.2	29.5	-99	-210	-5.1	64.6	90	-110	-193	-7.3	41.8	54.5	2.64	-1.2	0.3	0.9	2.8	9.33	6.6	1.9	5.0	3.2
b19	24	-2.1	1.2	3.8	4.4	172	39	9.9	18.6	24.4	-274	-16.1	21.9	44.5	44.2	-368	-21.5	27.4	51.1	53.3	5.36	-0.6	4.6	6.2	-3.8	1.88	6.7	1.9	4.9	3.0
s35932	7.6	24	-0.5	-1.2	4.5	22	-296	0	100	100	-54	353.7	139	230	224	-59	100	93.2	100	100	0.36	6	2.1	2	4.5	1.72	-1.4	2.9	5.8	5.8
s38417	7	14.7	1	-3	-2.2	0	0	0	0	0	122	-76.2	-17	1.6	-0.8	0	0	0	0	0	0.27	-1.8	1.5	3	1.8	1.56	-1.5	1.8	3.9	3.6
s38584	7.8	2.1	-4.7	-0.12	2.3	169	-14.8	33.1	-18.3	32.5	69.8	350.9	-108.2	-11.5	-25.4	0	0	-5.8*	0	0	0.4	1.2	3.8	5.5	-0.5	1.54	-1.3	4.8	12.2	10.4

References

[1] Yi-Hui Cheng,Yao-Wen Chang Integrating Buffer Planning with Floorplanning for Simultaneous Multi-Objective Optimization Asia and South Pacific Design Automation Conference 2004 (ASP-DAC'04) pp. 624-627

[2] W. Donath et al., `Transformational Placement and Synthesis.,*DATE'00*, pp. 194-201.

[3] L. N. Kannan, P. R. Suaris and H. G. Fang, .A Methodology and Algorithms for Post-Placement Delay Optimization., *DAC'94*, pp.327-332.

[4] A. H. Ajami and M. Pedram, .Post-Layout Timing-Driven Cell Placement Using an Accurate Net Length Model with Movable Steiner Points., *DAC'01*, pp. 595-600.

[5] M. Hrkic, J. Lillis and G. Beraudo, .An Approach to Placement-Coupled Logic Replication., *DAC'04*, pp. 711-716.

[6] C. W. Chang et al., .Fast Postplacement Optimization Using Functional Symmetries., *IEEE Trans. on CAD*, Jan. 2004, pp.102-118.

[7] S. C. Chang, L. P. P. P. van Ginneken and M. Marek-Sadowska,.Circuit Optimization by Rewiring., *IEEE Trans. on Computers*, Sep. 1999, pp. 962-969.

[8] C. Changfan, Y. C. Hsu and F. S. Tsai, .Timing Optimization on Routed Designs with Incremental Placement and Routing Characterization., *IEEE Trans. on CAD*, Feb. 2000, pp. 188-196.

[9] V. N. Kravets and P. Kudva, .Implicit Enumeration of Structural Changes in Circuit Optimization., *DAC'04*, pp. 438-441.

[10] A. Lu, H. Eisenmann, G. Stenz and F. M. Johannes, .Combining Technology Mapping with Post-Placement Resynthesis for Performance Optimization., *ICCD'98*, pp. 616-621.

[11] H. Vaishnav, C. K. Lee and M. Pedram, .Post-Layout Circuit Speed-up by Event Elimination., *ICCD'97*, pp. 211-216.

[12] K. Bernstein, C.-T. Chuang, R. Joshi, R. Puri, "Design and CAD challenges in sub-90nm technologies", *ICCAD* 2003.

[13] P. Saxena, N. Menezes, P. Cocchini, D. Kirkpatrick, "The scaling challenge: Can correct-by-construction design help?", *ISPD* 2003.

[14] H. B. Bakoglu. *Circuits, interconnections and packaging for VLSI*. Addison-Wesley, Reading, MA, 1990.

[15] J. Lillis, C. K. Cheng, and T. Y. Lin. Optimal wire sizing and buffer insertion for low power and a generalized delay model. *IEEE Journal of Solid-State Circuits*, 31(3):437–447, March 1996.

[16] C. J. Alpert, A. Devgan, and S. T. Quay. Buffer insertion for noise and delay optimization. In *Proc. of DAC*, pages 362–367, 1998.

[17] C. J. Alpert, M. Hrkic, J. Hu, and S. T. Quay. Fast and flexible buffer trees that navigate the physical layout environment. In *Proc. of DAC*, pages 24–29, 2004.

[18] P. Saxena, N. Menezes, P. Cocchini, and D. A. Kirkpatrick. Repeater scaling and its impact on CAD. *IEEE Trans. on CAD*, 23(4) : 451–463, April 2004.