Folded Circuit Synthesis: Logic Simplification Using Dual Edge-Triggered Flip-Flops

Inhak Han and Youngsoo Shin
Department of Electrical Engineering
KAIST, Daejeon 305-701, Korea

***Abstract*— Dual edge-triggered flip-flop (DETFF) captures data at both clock edges. We observe that conventional sequential circuit that contains single edge-triggered flip-flops (SETFFs) can be simplified by identifying pairs of combinational subcircuits that are structurally identical, removing one subcircuit of each pair, and providing input data twice by using DETFFs where SETFFs have been used. The resulting circuit is named folded circuit. We carry the observation to technology mapping problem, so that many identical subcircuits are synthesized early on in the design process. Experimental results with some test circuits indicate that circuit area is reduced as much as 16%.**

I. Introduction

A standard flip-flop is triggered only at one polarity of clock edge, rising or falling; it can thus be referred to as a single edge-triggered flip-flop (SETFF). Dual edge-triggered flip-flop (DETFF) [1], on the other hand, is triggered at both polarities. DETFF design can use half the clock frequency of SETFF counterpart for the same throughput, which greatly reduces clock power consumption. Designs that consist of multiple clock domains also benefit from DETFFs, e.g. 500 MHz blocks (in SETFF designs) now employ DETFFs and 250 MHz blocks continue to use SETFFs, while a single clock with source frequency of 250 MHz is distributed. DETFF circuit can be designed mostly by using conventional CAD tools, except that cares should be taken in timing analysis [2], clock gating [3], [4], and duty ratio variation.

In this paper, we employ DETFFs for completely different purpose. Given a sequential circuit that contains only SETFFs, the goal is to reduce circuit area by identifying pairs of identical subcircuits and dropping one subcircuit of each pair. DETFFs replace SETFFs in this process, where subcircuit receives input data from sequencing elements.

A. Motivational Example

Consider Fig. 1(a). Two sub-netlists within dotted boxes, N_1 and N_2, are structurally identical. Imagine that they are folded (or overlapped) as illustrated in Fig. 1(b). We modify standard DETFF to have two inputs: one input data is captured at rising edge of clock and the other is captured at falling edge. A primary input w is paired with an internal net w', and they are connected to a multiplexer, which is steered by clk; w and w' are now supplied at rising and falling edge, respectively.

The folded sub-netlist of Fig. 1(b) computes twice: when w, x, and y are supplied at rising edge of clock, and when w', x', and y' are supplied at falling edge. Note that tri-state buffers are inserted before G_1 and G_2, which are fanout gates

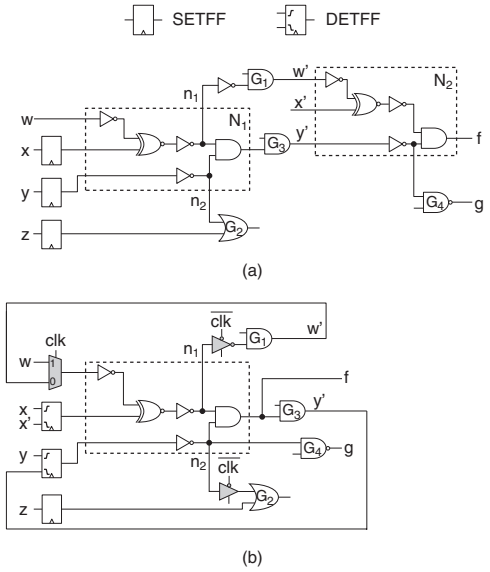

Fig. 1. A motivational example: (a) an original circuit using SETFFs and (b) a folded circuit.

of N_1 but not of N_2; this blocks the folded sub-netlist from propagating its outputs to G_1 and G_2 while clk is 0, when they are not supposed to compute. A tri-state buffer is not necessary before G_3 because its output is not loaded into flip-flop anyway when clk is 0.

II. Folded Circuit Synthesis

We address technology mapping problem to synthesize folded circuit. The input is a subject graph, which models combinational logic of a sequential circuit. In particular, we assume And-Inverter Graph (AIG) [5] $G = (N \cup \mathcal{I} \cup \mathcal{O}, E, w)$ as a subject graph, where a node $n \in N$ denotes a two-input AND, and $\mathcal{I}$ and $\mathcal{O}$ correspond to a set of primary inputs and primary outputs, respectively. A directed edge $e \in E$ has a binary weight on it which signifies the presence ($w(e) = 1$) or the absence of inverter ($w(e) = 0$). An example AIG for expression $f = ab'(c' + d) + cde$ is shown in Fig. 2(a); an edge with small circle indicates the existence of inverter.

The chance to discover identical subgraphs increases if we explore more than one subgraph for the same expression. We, thus, extend the basic AIG by modeling associative laws of AND operation [6]. In Fig. 2(a), cde is modeled by $(cd)e$;

978-1-4673-4743-3/13/$31.00 ©2013 IEEE

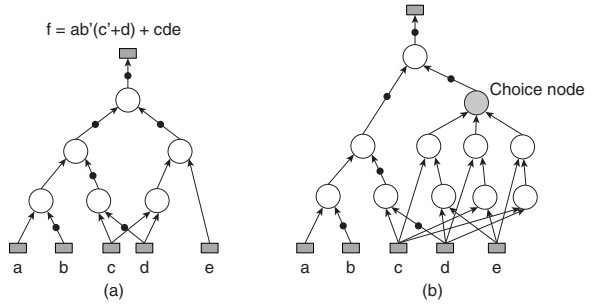

Fig. 2. (a) A basic AIG and (b) extended AIG, which includes modeling of associative laws of AND operation.

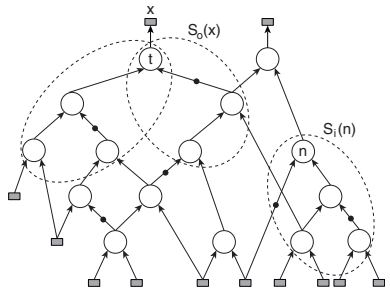

Fig. 3. Subgraphs of a subject graph.

both $c(de)$ and $(ce)d$ are also modeled in Fig. 2(b). A special node, named choice node, is employed to indicate that three subgraphs for cde are mutually exclusive.

A. Problem Definition

A primary output $x \in \mathcal{O}$ has a single incoming edge from a node $t \in N$. A subgraph $S_o(x)$ of G is a graph with t as a single root; a fanin cone of t that terminates at any nodes of N can be $S_o(x)$[1]. A subgraph $S_i(n)$ is a graph with $n \in N$ as a single root and the nodes with incoming edges from $\mathcal{I}$ as sources, i.e. $S_i(n)$ is a fanin cone of n and is uniquely determined. Fig. 3 illustrates the definition of subgraphs.

If $S_o(x)$ and $S_i(n)$ are isomorphic and disjoint, i.e. $S_o(x) \cap S_i(n) = \emptyset$, they are referred to as a *match* M; the number of vertices of $S_o(x)$ or $S_i(n)$ is called the size of a match and is denoted by $|M|$. In the folded circuit synthesis, we are only interested in disjoint matches. Two matches M_i and M_j are disjoint, if four subgraphs of them are mutually disjoint.

The ideal objective is to maximize $\sum |M_i|$, where all M_is are mutually disjoint. As we have seen in Fig. 1(b), some tri-state buffers are introduced after folding; their numbers should be minimized. Therefore, we set the practical objective as

$$\text{Maximize} \sum_i \left(|M_i| - \alpha\, b_i \right), \tag{1}$$

where b_i is the number of tri-state buffers required to implement M_i, and α is a weighting factor to account for the area difference of tri-state buffer and other logic gates.

[1]Strictly speaking, $S_o(x)$ is a set of subgraphs. Nevertheless, we keep using $S_o(x)$ to denote one of subgraphs for notational purpose.

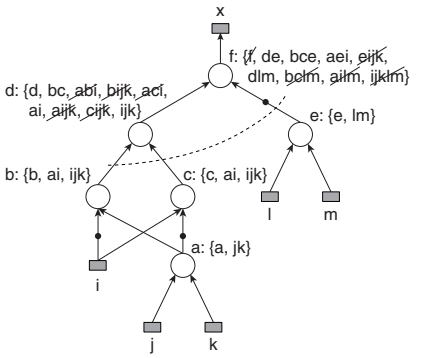

Fig. 4. An example to extract 3-feasible cut; a cut bce is shown as a dotted curve.

B. Synthesis Algorithm

The synthesis is performed in three steps: extracting all matches without regard to any overlap between them; selecting mutually disjoint matches with (1) as objective; and modifying the subject graph and performing technology mapping.

1) Extraction of Matches: Identifying $S_i(n)$, which is a fanin cone of n, can readily be done, because it is unique. Extracting $S_o(x)$ is governed by the size of itself we are interested in. We use k-feasible cut [7] for this purpose, where k corresponds to the maximum number of inputs of subgraphs we extract.

Consider Fig. 4, in which we try to extract $S_o(x)$ of 3-feasible cut. Depth-first search is performed starting from f, a node with outgoing edge to x, until we reach primary inputs (i, j, k, l, and m); each of them are assigned an implicit cut of itself, e.g. i is assigned a cut $\{i\}$. When we visit an internal node, its cut is made from itself and cross product of the cuts of its children, e.g. e is assigned $\{e, lm\}$. The cut of f constitutes 3-feasible cut after we remove f, $eijk$, $bclm$, $ailm$, and $ijklm$, which are not relevant. The cut and subgraph is one-to-one correspondence, e.g. cut bce shown as dotted line implies a subgraph made of nodes f and d. In our implementation, we extract 8-feasible cut at each primary output; 2-feasible cut is then dropped since smaller subgraphs are unlikely to help reduce circuit area; the decision on maximum cut size is experimentally addressed in Section III-B.

Once all subgraphs are identified, we compare S_o and S_i to find matches. Since the number of subgraphs is practically very large, we classify them and put them in different bins, so that comparison is made only within the same bin. This is illustrated in Fig. 5. The three subgraphs within dotted circles have the same numbers of nodes at each level, i.e. 1, 2, and 2, of edges that contain inverters, i.e. 2, and of nodes that have multiple fanouts, i.e. 1; they are placed in the same bin with S_o and S_i in different locations within the bin.

2) Selection of Matches: After all candidate matches are extracted, we need to pick the matches that are mutually disjoint with (1) as objective. This is a weighted set packing problem, which is NP-complete; all the nodes of a subject graph constitute a parent set, each subset contains the nodes

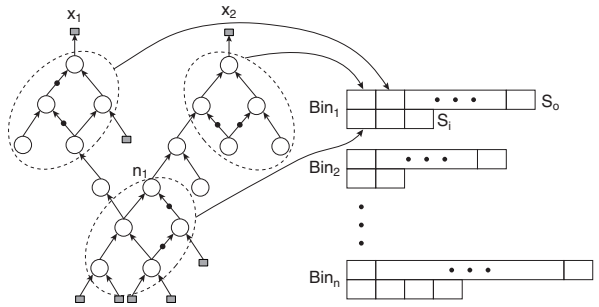

Fig. 5. Subgraph binning for fast match extraction.

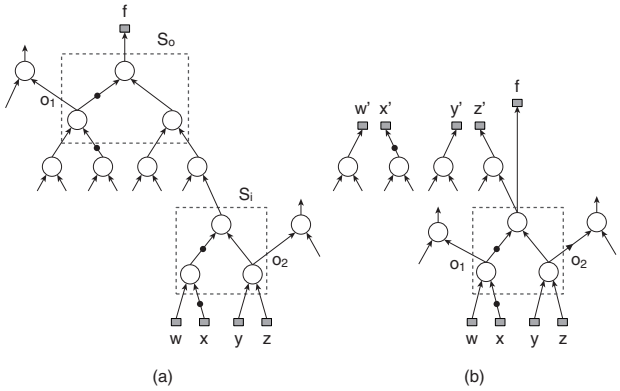

Fig. 6. (a) An original subject graph and (b) modified subject graph for technology mapping.

that belong to a particular match M_i, and a weight of each subset is set to $|M_i| - \alpha b_i$.

We develop a greedy heuristic to solve the problem. Each match is assigned a value

$$L(S_o)\,(|M_i| - \alpha b_i) \tag{2}$$

as a merit, where $L(S_o)$ is a level of the root of S_o, a subgraph that is matched by M_i. The level is assigned from primary inputs toward primary outputs of a subject graph. The subgraph S_o with larger $L(S_o)$ is less likely to have overlaps with other subgraphs, thus should be given higher priority. A match M_i with the largest merit is chosen. All the matches that are not disjoint with M_i are dropped from the list of candidates. The process repeats until the list becomes empty.

3) Technology Mapping: The original subject graph has to be modified before we perform technology mapping. Consider Fig. 6(a); assume that a pair of S_o and S_i is a match that we select. The subgraph S_o is first removed from the subject graph. Its outputs o_1 and f are re-connected to corresponding locations of S_i as shown in Fig. 6(b). The four inputs of S_o are labeled w', x', y', and z', and are assumed to be primary outputs during technology mapping. After mapping is complete, each of them is paired with corresponding input of S_i; e.g. w which is a flip-flop output is connected to DETFF with w' and z which is a circuit input is connected to a multiplexer with z'. Note that o_2 is an output of S_i but not S_o; it is marked and tri-state buffer is inserted after mapping.

Once the subject graph is modified, technology mapping is performed on each subgraph S_i one by one; the remaining nodes of the subject graph are then finally mapped.

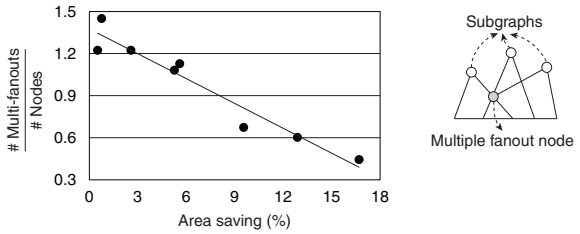

Fig. 7. Correlation between area saving and the number of multiple fanouts divided by the number of nodes of a subject graph.

III. Experimental Results

The folded circuit synthesis presented in Section II was implemented in SIS [8]. To assess the effectiveness of the synthesis, a set of sequential circuits was compiled from ISCAS and ITC benchmarks as well as from open cores [9], which are listed in Table I. A custom gate library was built for SIS in 32-nm commercial technology; it consists of 298 gates, which also includes the modified DETFF.

A. Area Reduction

The area of SETFF and folded circuits is compared in Table I. The circuits are ordered in increasing value of column 7, i.e. decreasing area saving of folded circuit over SETFF implementation. There is wide variation in area saving, which can be expected, i.e. saving is determined by inherent structure of a subject graph. But, it is a promising fact that there exist some circuits that achieve appreciable amount of saving, as much as 16%.

Area saving is determined by the amount of disjoint matches that are discovered. A node with multiple fanouts may be involved in the intersections among subgraphs that span that node, which is pictorially shown in Fig. 7; this is more likely to be the case as the number of fanouts increases. We count the number of all multiple fanouts (i.e. outgoing edges of multiple fanout nodes), then divide that number by total node count of a subject graph. This figure is shown in Fig. 7 for some circuits of Table I with area saving in x-axis. Strong negative correlation proves that our conjecture is true.

The number of nodes that are deleted from a subject graph ($\sum_i |M_i|$) does not directly correlate with area saving. For example of b15, 6.7% of nodes are deleted even though there is only 3% area saving. This is mainly because of extra tri-state buffers, e.g. 5% of gates are tri-state buffers in folded circuit of b15, which is why we consider (1) as the synthesis objective. In ps2, on the other hand, only 1.4% of nodes are deleted, which however yields 6% area saving. It turns out that many deleted nodes are multiple fanout nodes, which are not likely to be mapped to a single gate together with their adjacent nodes. Deleting them from a subject graph helps technology mapping in gate counts.

TABLE I

AREA COMPARISON OF SETFF (EQUIVALENT OF 2-INPUT NAND GATES) AND FOLDED CIRCUIT (NORMALIZED TO CORRESPONDING COLUMNS OF SETFF CIRCUIT); AREA IS MEASURED AS SUM OF STANDARD CELL AREAS

Name	SETFF circuit			Folded circuit			Runtime
	Comb.	FF	Total	Comb.	FF	Total	(s)
s5378	4632	712	5344	0.79	1.14	0.84	3
usb_phy	2329	440	2769	0.79	1.15	0.85	3
s1423	2938	325	3263	0.86	1.15	0.89	1
s9234	4579	553	5132	0.87	1.13	0.89	2
wb_dma	22579	2348	24927	0.89	1.09	0.91	21
usb_funct	51908	7276	59184	0.91	1.09	0.94	431
wb_conmax	53160	3385	56545	0.92	1.23	0.94	232
ps2	9162	788	9950	0.93	1.13	0.94	19
s15850	15250	1970	17220	0.92	1.12	0.95	28
ac97	32880	7750	40630	0.94	1.03	0.95	121
b17	107284	5927	113211	0.96	1.09	0.96	338
mem_ctrl	34643	4729	39372	0.96	1.06	0.97	33
b15	35141	1876	37017	0.97	1.08	0.97	21
aes_core	152838	2214	155052	0.98	1.10	0.98	200
s38417	87769	6562	94331	0.98	1.05	0.98	262

B. Runtime

Synthesis runtime is reported in the last column of Table I. Clearly, more time is spent for a circuit of larger combinational gates, since bigger subject graph is processed. In most circuits, about 50% of runtime is spent in extraction of matches, about 5% in selection of matches, and 45% in technology mapping. Therefore, we can roughly state that folded circuit synthesis in current implementation takes twice the time of standard technology mapping.

The selected matches are less than 1% of total extracted matches in most circuits, e.g. 0.35% (21 out of 6041) in s5378. The two steps may be integrated for the benefit of smaller runtime.

1) Maximum Cut Size: Maximum cut size affects area saving and synthesis runtime. As we increase its value, the maximum size of subgraphs we search for match increases, which offers a chance of more area saving, but it comes at the cost of larger runtime. Fig. 8(a) shows area saving of three sample circuits while we vary maximum cut size. Area saving tends to saturate as we increase maximum cut size; this is understandable because larger subgraph S_o has a smaller chance to be matched to a subgraph S_i. We notice from Fig. 8(b) that runtime increases substantially beyond maximum cut size of 8; this fact, together with the observation from Fig. 8(a), guides us to fix the maximum cut size to 8 in our implementation.

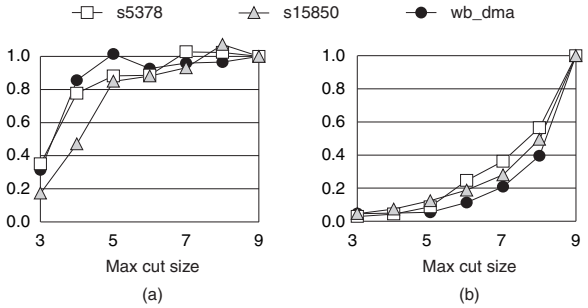

Fig. 8. Folded circuit synthesis with varying maximum cut size: (a) area saving and (b) runtime.

IV. CONCLUSION

We have introduced a concept of folded circuit. The synthesis problem has been formulated as a part of technology mapping process. The key idea is to discover pairs of isomorphic subgraphs, so that one subgraph of each pair can be removed for the benefit of circuit area. The sub-circuit corresponding to remained subgraph is supplied input data twice in a single clock period, which is made possible by using DETFFs. The folded circuit achieves area saving as much as 16% in the circuits we tested.

Circuit timing is more complex in folded circuit; this should be taken into account during the synthesis, which is left for future investigation. A simple heuristic algorithm was deployed for selection of matches; more elaborate algorithm would result further area saving and thus should be developed.

REFERENCES

[1] S. Unger, "Double-edge-triggered flip-flops," *IEEE Trans. Comput.*, vol. C-30, no. 6, pp. 447–451, June 1981.
[2] C. Oh, S. Kim, and Y. Shin, "Timing analysis of dual-edge-triggered flip-flop based circuits with clock gating," in *Proc. Int. Conf. Integr. Circuits Des. Tech.*, May 2009, pp. 59–62.
[3] R. Llopis, "Electronic circuit with dual edge triggered flip-flop," U.S. Patent 6 137 331, Oct. 24, 2000.
[4] J. Tschanz, D. Somasekhar, and V. De, "Gating for dual edge-triggered clocking," U.S. Patent 7 109 776, Sept. 19, 2006.
[5] A. Kuehlmann and F. Krohm, "Equivalence checking using cuts and heaps," in *Proc. Des. Autom. Conf.*, June 1997, pp. 263–268.
[6] E. Lehman, Y. Watanabe, J. Grodstein, and H. Harkness, "Logic decomposition during technology mapping," in *Proc. Int. Conf. Comput.-Aided Des.*, Nov. 1995, pp. 264–271.
[7] S. Chatterjee, A. Mishchenko, and R. Brayton, "Reducing structural bias in technology mapping," in *Proc. Int. Conf. Comput.-Aided Des.*, Nov. 2005, pp. 519–526.
[8] E. Sentovich *et al.*, "SIS: a system for sequential circuit synthesis," UC Berkeley, Tech. Rep. UCB/ERL M92/41, May 1992.
[9] OpenCores. [Online]. Available: http://www.opencores.org/