

# Combinational Circuits with Feedback

*Abstract of Ph.D. Dissertation*

Marc D. Riedel  
Caltech 136–93, Pasadena, CA 91125  
riedel@caltech.edu

## SUBMISSION

**Ph.D. Dissertation:** in Electrical Engineering at the California Institute of Technology, advisor Prof. Jehoshua Bruck; degree expected June 11, 2004.

**Supporting Paper:** “The Synthesis of Cyclic Combinational Circuits,” DAC, Anaheim, 2003, pp. 163–168 (received the *Best Paper Award*)

## 1. INTRODUCTION

Digital circuits are called *combinational* if they are memoryless: they have outputs that depend only on the current values of the inputs. A common misconception is that combinational circuits cannot contain cycles; that is to say, they must be designed without any feedback paths. Consider the circuit shown in Figure 1. It computes three output functions,  $f_1$ ,  $f_2$  and  $f_3$ , of three input variables  $x_1$ ,  $x_2$  and  $x_3$ . The corresponding equations are:

$$f_1 = \bar{x}_1 f_3 + \bar{x}_2 \bar{x}_3, \quad (1)$$

$$f_2 = x_1 \bar{f}_1 + \bar{x}_1 \bar{x}_2 \bar{x}_3, \quad (2)$$

$$f_3 = \bar{x}_3 \bar{f}_2 + \bar{x}_2 x_3. \quad (3)$$

Due to the feedback path through gates  $g_1$ ,  $g_3$ ,  $g_4$ ,  $g_6$ ,  $g_7$  and  $g_9$ , there is a cyclic dependency:  $f_2$  depends on  $f_1$ ,  $f_3$  depends on  $f_2$ , and  $f_1$  depends on  $f_3$ . Nevertheless, this circuit is combinational. To see this:

- Suppose that  $x_1 = 0$ . In this case,  $f_2 = \bar{x}_2 \bar{x}_3$ , and so  $f_3 = x_2 \bar{x}_3 + \bar{x}_2 x_3$ , and so  $f_1 = \bar{x}_2 + \bar{x}_3$ .
- Suppose that  $x_1 = 1$ . In this case,  $f_1 = \bar{x}_2 \bar{x}_3$ , and so  $f_2 = x_2 + x_3$ , and so  $f_3 = \bar{x}_2$ .

With  $x_1 = 0$ , the feedback path is broken at gate  $g_4$ ; with  $x_1 = 1$ , it is broken at gate  $g_1$ . For any assignment of values to the input variables, all outputs are determined regardless of the prior values on the wires and independently of all timing assumptions. We might say that there is feedback in a *topological* sense, but not in an *electrical* sense.

The circuit implements the functions

$$f_1 = \bar{x}_1 \bar{x}_2 + \bar{x}_1 \bar{x}_3 + \bar{x}_2 \bar{x}_3, \quad (4)$$

$$f_2 = x_1 x_2 + x_1 x_3 + \bar{x}_1 \bar{x}_2 \bar{x}_3, \quad (5)$$

$$f_3 = x_1 \bar{x}_2 + \bar{x}_2 x_3 + \bar{x}_1 x_2 \bar{x}_3. \quad (6)$$

Note that the circuit consists of 9 two-input AND/OR gates (as well as inverters). In contrast, the best acyclic circuit implementing these three functions requires 10 such gates (as well as inverters).

In this dissertation, we demonstrate that feedback in combinational logic is not a mere theoretical curiosity: rather, it is an idea with general applicability, offering opportunities for significant optimizations.

## 2. PRIOR WORK

In an earlier era, theoreticians commented on the possibility of having cycles in combinational logic, and conjectured that this might be a useful property [1], [2]. Most notably, Rivest presented a family of cyclic circuits with a striking property: in an asymptotic sense, each is two-thirds the size of the smallest possible equivalent acyclic circuit [3].

In a later era, practitioners observed that cycles sometimes appear in combinational circuits synthesized from high-level descriptions [4]. In such examples, feedback either is inadvertent or else is carefully contrived. For instance, occasionally it is introduced during resource-sharing optimizations at the level of functional units. Since logic synthesis and verification tools balk when given cyclic circuits, methods were proposed for analyzing such designs [5], [6], and untangling them [7].

Although the premise of cycles in combinational logic is well-established, no one has attempted the synthesis of such circuits.

## 3. CONTRIBUTIONS

We present a general methodology for the synthesis of multi-level circuits with feedback. Optimizations targeting area and delay are introduced in an iterative fashion at various points in the design flow. Efficient analysis techniques play a central role.

### 3.1 Analysis

We describe a framework for exact (i.e., false-path aware) gate-level timing analysis, based on efficient symbolic techniques, both BDD-based and SAT-based. Our algorithm identifies the true delay of *sensitizable* paths. If there exist sensitizable paths of “infinite” length (i.e., paths that form cycles), then the circuit is deemed non-combinational.

In the example of Figure 1, there are two sensitizable paths of length 6:

- If  $x_1 = 1$  and  $x_3 = 0$ , then the path through gates  $g_1, g_3, g_4, g_6, g_7$  and  $g_9$  is sensitized.
- If  $x_1 = 0$  and  $x_3 = 0$ , then the path through gates  $g_4, g_6, g_7, g_9, g_1$  and  $g_3$  is sensitized.

Other input combinations yield shorter sensitized paths.

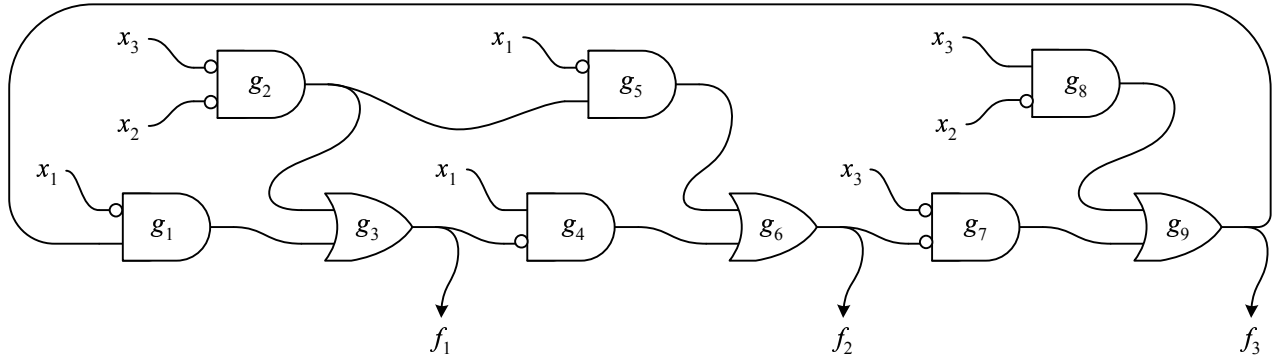


Figure 1: A cyclic combinational circuit.

### 3.2 Synthesis

Our approach for synthesis is conceptually very general. Feedback is introduced in the re-structuring / minimization phase, optimizing a circuit description for area while satisfying delay constraints. The cyclic optimizations are carried through to the decomposition and technology mapping phases.

A branch-and-bound search is performed on the space of possible *cyclic dependencies*. The search either begins with a dense edge set and prunes it, or begins with a sparse edge set and augments it, until a combinational solution is found. We have explored a variety of heuristics to expedite the search. For instance, we can prioritize progress slightly at the expense of quality. Also, we can limit the density of edges a priori or prune the set of edges incrementally.

For the target functions in Equations (4) – (6), three possible configurations are shown in Figure 2. Choices (a) and (b) do not result in combinational solutions, whereas choice (c) does.

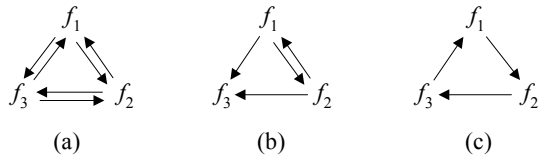


Figure 2: Search space of cyclic dependencies.

### 3.3 Results

We have implemented our methodology in a program called CYCLIFY, integrated within the Berkeley SIS environment. The results of trials indicate that cyclic solutions are not a rarity; they can readily be found for most networks that are not trivially simple or sparse. In Figure 3, we present a simple area comparison for common benchmark circuits. The substitution/minimization operation is performed with the `simplify` command in the Berkeley SIS package.

## 4. DISCUSSION

This dissertation should help dispel any notion that the terms “combinational” and “acyclic” are synonymous. On a theoretical front, we have produced a family of cyclic circuits that are asymptotically one-half the size of the smallest equivalent acyclic circuits. On a practical front, our synthesis results are sufficiently convincing to augur a paradigm shift in circuit design: nearly all combinational circuits can be optimized with feedback.

Logic Synthesis Benchmarks				
	SIS Simplify	CYCLIFY	Improvement	mins:secs
p82	104	90	13.5%	02:03
t4	109	89	18.3%	00:02
apla	185	131	29.2%	00:31
tms	185	158	14.6%	01:17
m2	231	207	10.4%	06:02
t1	273	206	24.5%	21:40
exp	320	260	18.8%	33:26
in2	397	291	26.7%	00:45
ex6	85	76	10.6%	00:06
bbsse	118	106	10.2%	00:08
sse	118	106	10.2%	00:10
5xp1	123	109	11.4%	00:01
s386	131	113	13.7%	00:08
cse	212	177	16.5%	00:05
s510	260	227	12.7%	00:05
ex1	309	276	10.7%	09:11

Figure 3: Cost (literals in factored form) of Berkeley SIS Simplify vs. CYCLIFY for benchmarks.

In future work, we plan to extend our cyclic methodology to higher-level and lower-level aspects of synthesis, e.g., up to system-level/behavioral-level designs and down to physical synthesis.

## 5. REFERENCES

- [1] W. H. Kautz, “The Necessity of Closed Circuit Loops in Minimal Combinational Circuits,” *IEEE Trans. Comp.*, Vol. C-19, pp. 162–166, 1970.
- [2] D. A. Huffman, “Combinational Circuits with Feedback,” *Recent Developments in Switching Theory*, A. Mukhopadhyay, ed., pp. 27–55, 1971.
- [3] R. L. Rivest, “The Necessity of Feedback in Minimal Monotone Combinational Circuits,” *IEEE Trans. Comp.*, Vol. C-26, No. 6, pp. 606–607, 1977.
- [4] L. Stok, “False Loops Through Resource Sharing,” *Int’l Conf. Computer-Aided Design*, pp. 345–348, 1992.
- [5] S. Malik, “Analysis of Cyclic Combinational Circuits,” *IEEE Trans. Computer-Aided Design*, Vol. 13, No. 7, pp. 950–956, 1994.
- [6] T. R. Shiple, “Formal Analysis of Synchronous Circuits,” Ph.D. Dissertation, University of California, Berkeley, 1996.
- [7] S. A. Edwards, “Making Cyclic Circuits Acyclic,” *Design Automation Conf.*, pp. 159–162, 2003.
- [8] M. Riedel and J. Bruck, “Cyclic Combinational Circuits: Analysis for Synthesis,” *Int’l Workshop Logic and Synthesis*, pp. 105–112, 2003.
- [9] M. Riedel and J. Bruck, “The Synthesis of Cyclic Combinational Circuits,” *Design Automation Conf.*, pp. 163–168, 2003.