Microelectronics Journal 45 (2014) 1212–1219

Contents lists available at ScienceDirect

Microelectronics Journal

journal homepage: www.elsevier.com/locate/mejo

Methods for automated detection of plagiarism in integrated-circuit layouts

Dominik Kasprowicz*, Hilekaan Wada

Institute of Microelectronics and Optoelectronics, Warsaw University of Technology, ul. Koszykowa 75, 00-662 Warsaw, Poland

ARTICLE INFO

Article history:
Received 6 December 2013
Received in revised form
7 March 2014
Accepted 7 April 2014
Available online 6 May 2014

Keywords:
Plagiarism detection
Copyright protection
Integrated circuit layout
Layout matching

ABSTRACT

Student projects have always been plagued by plagiarism. Integrated-circuit (IC) design courses are no exception. Since layout is considered the most laborious part of circuit design, it is common for students to reuse their colleagues' work with some minor modifications intended to make the cheating harder to detect. While software detecting plagiarism in text or computer code is commonly used these days, no counterpart exists for IC layouts. This paper proposes several criteria of IC-layout dissimilarity that can be used for computer-aided layout matching. A program based on these criteria is shown to successfully identify similar layouts in a pool of designs.

1. Introduction

Academic courses in integrated-circuit (IC) design usually include projects where a student is asked to design a simple IC-cell down to the physical layout level. As the circuit must be simple enough to be designed by a relatively unexperienced person, the number of architectures useful for that purpose is quite limited. Therefore, it is natural for some circuits, like the Miller transconductance amplifier or a flip-flop, to be reused year over year or even within a single student group. The design of the physical layout of a given circuit is usually viewed by students as the most challenging – or at least the most laborious – part of the assignment. As a consequence, some students are tempted to reuse their colleagues' work, usually with minor modifications intended to make the cheating harder to detect. While it is relatively easy for an instructor to spot cases of plagiarism within a single student group, reuse of layouts created a couple of years back is likely to go unnoticed. Thus, a computer application pointing the user to "suspiciously similar" items in a layout repository would be a useful tool. Whereas software supporting detection of plagiarism in essays or computer code is already mature and widely used (see e.g. [1] or [2] for an overview of currently used methodologies), no counterpart for circuit layouts is available. To the best of the authors' knowledge, no results of academic studies in this field have been published either.

This paper summarizes the authors' initial efforts toward automation of detection of IC-layout plagiarism. It is an extended version of the work presented in [3]. Section 2 contains general remarks on the problem of layout comparison. Formal measures of layout dissimilarity are proposed in Section 3. Section 4 contains thorough analysis of the outcome of an automated scan for plagiarism performed on a given pool of layouts. The section starts with a discussion of results obtained for the set of eight layouts presented in Fig. 7 at the end of this paper, which permits easy understanding of the source of strengths and weaknesses of each measure of dissimilarity. Then, results for a much larger set of layouts are presented to draw more general conclusions regarding those measures. The section ends with an analysis of the impact of the number of layouts and their complexity on the runtime of the plagiarism detection procedure. Section 5 summarizes the work.

2. IC layout matching – general observations

An important step toward creation of a layout-plagiarism detector is *verbalization* of what layouts are considered similar. Definitely, similar sizes of corresponding components, e.g. the input differential pair, are neither necessary nor sufficient to decide that one layout is a copy of another. Indeed, component sizes are strongly influenced by the project specifications like power consumption, area or performance. If several assignments share a common architecture, the most obvious way to discourage the students from identically sizing their transistors and passives is imposing different specifications on each design. This policy, however, does not prevent the students from copying the physical

* Corresponding author. Tel.: +48 222347207.
E-mail addresses: D.Kasprowicz@imio.pw.edu.pl, dkasprow@imio.pw.edu.pl (D. Kasprowicz).

http://dx.doi.org/10.1016/j.mejo.2014.04.023
0026-2692/© 2014 Elsevier Ltd. All rights reserved.

layout of the circuits designed by their colleagues, although some effort must be invested in resizing the components. Thus, other similarity measures must be sought.

Two comparison criteria are contemplated: one is the location of transistors, the other is the shapes of the interconnects between those transistors. Of course, similarity of transistor placement and similarity of interconnect routing are still vague concepts that need further formal definitions. Such definitions are proposed in Section 3. Preferably, a tool comparing two layouts should be fairly insensitive to basic layout transformations: scaling, rotation, and reflection. This requirement gains importance if the transistor sizing and the placement of terminals are the designer's choice. In such a situation, resizing an existing layout (perhaps also changing its aspect ratio) along with a rotation or reflection is a simple way of making the layouts dissimilar at first glance. While detecting a rotated and/or reflected copy of a given design is conceptually trivial, detecting similarities between two layouts of a substantially different size or aspect ratio is trickier. Since design rules impose numerous upper and lower limits on the sizes of layout features and distances between them, a resized design is never just a magnified or shrunk copy of the original. Thus, even size normalization is not guaranteed to make the two layouts identical.

Another problem stems from the fact that designs sharing the same architecture may differ in the number of transistors, i.e. MOSFET fingers. In fact, such differences may occur even in cases where one layout is an almost exact copy of another. Such a situation takes place in Designs A, B, and C (see Fig. 7), whose top-right portions differ substantially in the number and location of transistors, while the rest remains almost identical. From the algorithmic point of view, if two sets or sequences must be compared elementwise, the difference in their sizes always introduces some degree of ambiguity. Still, software detecting plagiarism must handle such situations.

Detecting plagiarism in student assignments requires comparing a large number of relatively simple layouts stored in the teacher's repository. Detecting plagiarism in a pool of L designs requires $L(L-1)/2$ comparisons, which might be costly in terms of computation time. Thus, it might be tempting to split this task into two stages. First, some small set of characteristic properties (a “signature”) would be extracted from each layout. This extraction procedure would only run L times, so even computationally expensive algorithms would be allowed at this stage. The second step would involve pairwise comparisons of those signatures, being relatively small datasets. Even though the overall time complexity of the whole procedure remains $O(L^2)$, such partitioning of the task might bring substantial speedup.

The task of detecting IC-layout plagiarism is much different from detecting plagiarism in essays or computer code. The latter problems are inherently one-dimensional since they involve comparing strings of characters. An IC layout, on the other hand, is a two-dimensional (and multilayered) entity. Theoretically, IC-layout comparison could rely on bitmap-analysis algorithms. Such an approach, however, would be extremely a wasteful of computational resources. Image-comparison algorithms usually begin their operation by performing either feature extraction or some dimensionality-reduction procedure, e.g. the Principal-Component Analysis. The actual comparison is subsequently performed on such a reduced set of crucial attributes rather than on the full set of pixels. The computer representation of an IC layout already contains the crucial data, i.e. the coordinates of polygon vertices, and can be easily analyzed to extract parameters like polygon centroids, moments of area, or other measures described later in this paper. Thus, transforming such a vector representation into a bitmap would be an unnecessary (and very costly) step back. Nevertheless, some measures known from image analysis, like moment invariants, can be easily adapted to handle shapes represented as sequences of vertices.

Vector representation has an additional advantage of enabling fine-grained comparison (e.g. polygon-by-polygon) rather than comparing bitmaps representing entire circuits. This is important because, as mentioned before, similar placement and routing of corresponding components is usually indicative of plagiarism, irrespective of any difference in the dimensions of those components. If the two layouts were to be “blindly” compared as bitmaps, those size differences would likely blur the existing similarities between those layouts.

3. Layout dissimilarity measures

The following definitions and assumptions are used throughout this paper. A *layout signature* is a sequence of numbers describing such properties of the underlying layout that a large difference (however defined) between two signatures implies strong difference between the underlying layouts. That difference between signatures will be referred to as *dissimilarity measure*. A *layout comparator* is a computer application calculating such dissimilarity measures within a given set of layouts. The word *transistor* denotes a MOSFET channel, defined as the intersection of lithographic masks of polysilicon and the active layer (diffusion). If a MOSFET channel is divided into multiple fingers, each of them is treated in this work as a separate transistor unless noted otherwise. The term *transistor coordinates* or *location* applies to the centroid of the transistor's channel. Sections 3.1–3.3 outline proposed dissimilarity measures based on transistor location.

3.1. Transistor distance to the layout center – TDLC

This method analyzes the distances d of all the N transistors to the center of the bounding box of all the transistor centers as illustrated in Fig. 1. Thus, the signature is simply a sequence of those distances arranged in non-ascending order

$$\{d_1, d_2, \ldots, d_N\}, \quad d_1 \geq d_2 \geq \cdots \geq d_N. \tag{1}$$

This leads to the following measure of dissimilarity between designs A and B

$$TDLC(A,B) = \frac{1}{M}\sqrt{\sum_{m=1}^{M}(d_{Am} - d_{Bm})^2} \tag{2}$$

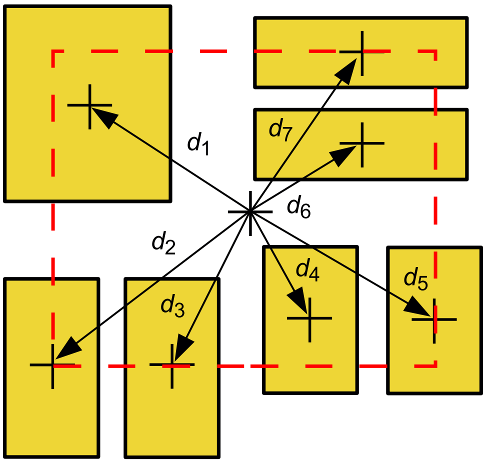

Fig. 1. Illustration of the definition of the distances d of transistors from the layout center. The dashed contour outlines the box bounding all the transistor centroids (black crosses). Its center is the reference point.