

A Systolic Design-Rule Checker

RAJIV KANE AND SARTAJ SAHNI, SENIOR MEMBER, IEEE

Abstract—We develop a systolic design-rule checker (SDRC) for rectilinear geometries. This SDRC reports all width and spacing violations. It is expected to result in a significant speed up of the design-rule check phase of chip design.

Keywords and Phrases: Design-Rule Checks, feature width, spacing, rectilinear geometries, systolic systems.

I. INTRODUCTION

RAPID ADVANCES in technology are making it possible to fabricate circuits of an ever increasing complexity. This increase in circuit complexity poses a severe challenge to the algorithms presently in use in design automation tools. One of the ways to meet the challenge is to develop new computer architectures capable of running these design automation algorithms efficiently. Another approach is to develop faster algorithms.

Several new architectures and corresponding algorithms have recently been proposed for design automation. Blank *et al.* [2] describe a bit-map processor architecture suitable for Boolean operations, wire routing using Lee's algorithm, and for some design-rule check (DRC) functions such as shrink and expand. Mudge *et al.* [7] describe a Cytocomputer architecture adapted for DRC and Lee-type wire routing. Yet another DRC architecture is described in [11]. Some other references for special purpose architectures and associated algorithms for wire routing are [3] and [8]. A parallel-processing approach for logic module placement has been developed by Ueda *et al.* [13]. Simulation has also been the focus of several new architectural studies. The most popular such development is the Yorktown Simulation Engine [10], [3], [5]. Another logic simulation machine is described by Abramovici *et al.* [1].

In this paper, we shall be concerned with the design of a systolic system for design-rule checks. Our design differs from all earlier work on special-purpose architectures for design automation in that ours is the first systolic design. Of course, systolic designs have been studied for quite some time. A valuable reference is [6]. Our systolic system for DRC's differs from earlier work on hardware assisted DRC's in that it is edge-based rather than bit-map-based. Consequently, it has the potential of being much faster and less expensive than earlier designs.

Manuscript received August 1, 1983; revised July 15, 1986. This research was supported in part by the Office of Naval Research under Contract N00014-80-C-0650 and in part by the Microelectronics and Information Sciences Center at the University of Minnesota.

R. Kane is with Daisy Systems, San Jose, CA 95127.

S. Sahn is with the University of Minnesota, Minneapolis, MN 55455. IEEE Log Number 861127.

Specifically, our systolic design-rule checker (SDRC) checks for spacing and width errors. The design may be extended to include other design-rule checks. Our design points out the potential for systolic systems in design automation applications.

II. POLYGONS AND ERRORS

In arriving at our SDRC, we made several assumptions on the nature of the polygons to be handled and also on the type of errors to be checked for. First, we assume that polygons are composed of horizontal and vertical edges only. Hence, only right-angled bends are permitted. Polygons may contain holes. These holes are also restricted to be polygons with right-angled bends. Fig. 1 shows two example polygons that satisfy these restrictions.

This restriction on the edges composing a polygon allows a compact representation of each polygon. This representation consists of the following.

1) *Polygon number:* Each polygon is assigned a unique number. Holes within a polygon are assigned the same number as the enclosing polygon.

2) *A sequence of polygon vertices:* This sequence begins at the lowermost left-hand vertex of the polygon and is obtained by traversing the polygon so that its interior lies to the left of the edge being traversed. Since all edges are either horizontal or vertical, the polygon vertices (except the first) may be described by providing a single coordinate. Thus, the polygon of Fig. 1(a) is represented as

$$p, n, x_1, y_1, x_2, y_3, x_4, y_5, x_6, y_7, x_8, y_1.$$

The first symbol p identifies this as an enclosing polygon; n is the polygon number. In case of a hole, an h is used in place of the p . Holes are traversed such that the interior is to the left of each edge traversed. The representation for the polygon and holes of Fig. 1(b) is

$$p, n, x_1, y_1, x_2, y_3, x_4, y_5, x_6, y_7, x_8, y_9, x_{10}, y_{11}, x_{12}, y_1$$

$$h, n, x_{13}, y_{13}, x_{14}, y_{15}, x_{16}, y_{17}, x_{18}, y_{19}, x_{20}, y_{13}$$

$$h, n, x_{21}, y_{21}, x_{22}, y_{23}, x_{24}, y_{25}, x_{26}, y_{21}.$$

The SDRC assumes that the polygons are well formed. Specifically, open polygons (Fig. 2(a)), polygons with shared edges (Fig. 2(b)), polygon overlaps (Fig. 2(c)), and polygons sharing an edge with a hole (Fig. 2(d)) are not permitted. While this assumption of well-formedness is not essential to our discussion, it enables us to concentrate on spacing and width issues. A minor modification to our design allows the SDRC to check for the above

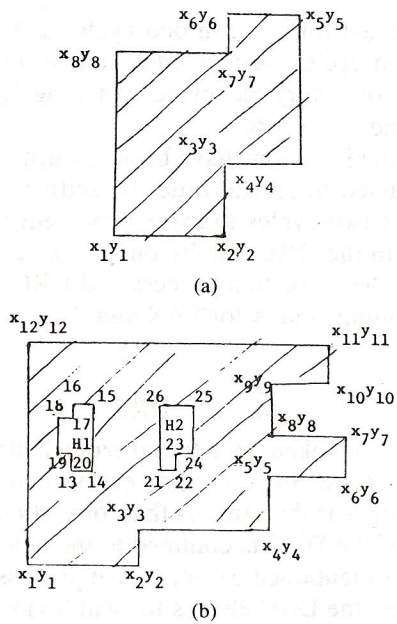


Fig. 1. Examples of polygons. (a) No holes. (b) Two holes H1 and H2. Shaded area is the interior of the polygon.

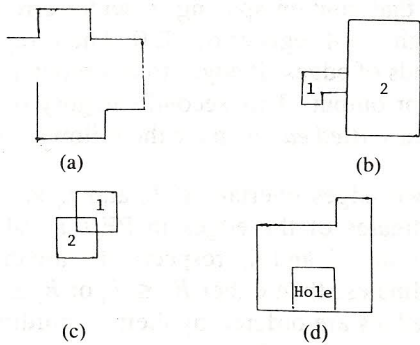


Fig. 2. Malformed polygons.

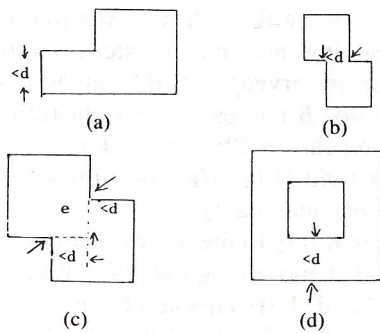


Fig. 3. Polygons with width errors.

malformations. Also, these inconsistencies need to be explicitly checked before one can apply bit-map-based width and spacing checks.

Let w denote the minimum allowable feature width. Fig. 3 gives examples of polygons with width error. Many designers do not regard Fig. 3(c) as an error unless the distance e is less than w . Our SDRC is easily changed to account for this variation. The only change needed is to compare w with e rather than d .

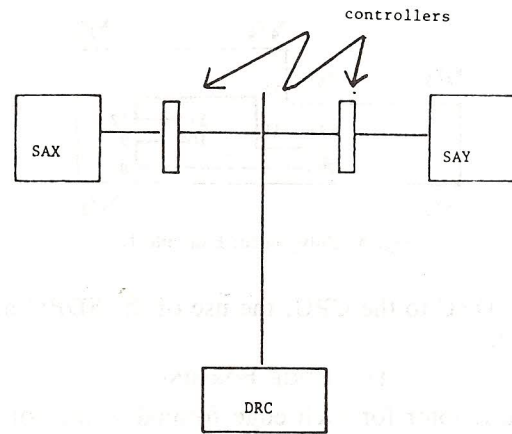


Fig. 4. SDRC architecture.

III. SDRC ARCHITECTURE

The SDRC is a hardware device that may be attached to a computer system as a peripheral or directly to the CPU as in the case of a floating-point processor. A block diagram of the SDRC appears in Fig. 4. The major components of an SDRC are two systolic sort arrays (SAX and SAY), controllers for these sort arrays, and a systolic design-rule checker (DRC). Note that we use SDRC to denote the entire systolic design-rule check system of Fig. 7 and DRC to refer to a component of SDRC that performs the actual design-rule checks. This component is also systolic in nature. When design-rule checks are to be performed, the CPU sends the compact descriptions of the polygons to the SDRC. This description is transformed into explicit edges by the controllers for SAX and SAY. Horizontal edges are created by the controller for SAX and inserted into SAX. Vertical edges are formed by the controller for SAY and inserted into SAY. The sort arrays sort the edges into lexical order. Thus, the SAX sorts edges by y -coordinates and within y -coordinates by x -coordinates. Recall that we have assumed that there are no overlapping edges. So, even though every horizontal edge has two x -coordinates, there is a unique lexical ordering for the horizontal edges. Similarly, there is a unique ordering for the vertical edges.

As we shall see in the next section, the SAX and SAY are simply systolic priority queues. Consequently, as soon as the edges have been formed and entered into the SAX and SAY, they may be transmitted in lexical order to the DRC. First, SAX sends its edges to the DRC, which examines them for width violations in the y -direction and spacing violations in the x -direction. All detected errors are transmitted back to SAX. Next, SAY transmits its edges to the DRC, which examines them for width errors in the x -direction and spacing errors in the y -direction. These errors are sent back to SAY. The errors collected in SAX and SAY may then be communicated back to the CPU.

Clearly, by using two DRC's, the horizontal and vertical edge processing may be effectively overlapped. Further, by providing a data path for the errors to go directly

