Linear Algorithms For Two CMOS Layout Problems

Rolf Müller
Thomas Lengauer
Fachbereich 17
University of Paderborn
4790 Paderborn
West Germany

ABSTRACT

[UC,BNR] formulate a linear layout problem for static CMOS gates and give partial solutions of the problem. [O] reformulates the problem in two ways for dynamic CMOS cells and gives partial solutions. We give complete solutions to both problems formulated by [O] by giving linear algorithms.

1. Introduction

In bulk CMOS there is a certain widely accepted way of implementing complex (static or dynamic) gates. In this method the channels of n- and p-transistors are laid out in rows, the p-transistors above the n-transistors. Transistor gates are run vertically such that above each n-transistor there is the corresponding p-transistor which is gated by the same input (see Figure 1). This layout style allows for a dense layout with efficiently designed wells while preventing latchup phenomena.

[UC] formulate a graph theoretic problem for minimizing the width of layouts of complex static gates in bulk CMOS using this layout method. They provide a heuristic for minimizing layout width and leave open the question whether the problem can be solved exactly in an efficient manner. [BNR] give a linear time solution for a restricted class of CMOS gates (those having dual Eulerian paths). [O] gives two reformulations of the graph problem for dynamic CMOS gates (cascodes) which are used in the Yorktown Silicon Compiler [BB]. He also gives linear time algorithms for a restricted class of circuits (those having Eulerian paths). In this paper we extend the results of [O] by giving a linear time solution of both of the problems he formulates.

The paper is organized as follows: Section 2 gives an introduction to the technological issues of the topic. Section 3 defines the two layout problems. Section 4 solves the first layout problem. Section 5 solves the second layout problem. Section 6 compares the two layout problems. Section 7 gives conclusions.

2. Technological considerations

[O] discusses the following design of complex dynamic CMOS gates (Figure 2). The circuit operates in two phases. In the first phase PC is high and the capacitance C_S is precharged to 1 over the p-transistor T_1. The n-transistor T_2 is turned off and thus prevents C_S from being discharged. By the end of the first phase the inputs gating the pulldown network are stable. In the second phase PC is low and C_S can be discharged through the pulldown network if the current assignment to the inputs opens a path to T_2. After the second phase out is 1 exactly iff C_S has been discharged through the pulldown network.
We are interested in the layout of the pulldown network. Here we will only consider series-parallel networks. Series-parallel networks occur most often in practice and correspond directly to Boolean formulas with the series connection implementing logical-and and the parallel connection implementing logical-or. Furthermore, series-parallel networks are considerably easier to analyse than general networks.

We will layout the n-transistors of the pulldown network in a row. Here two adjacent transistors can be put closer to each other if they share a channel terminal than if they don't. This is because disconnected channel terminals of adjacent transistors must be separated electrically either through sufficiently large substrate regions or through permanently cut-off transistors. In order to minimize the width of the layout we therefore have to minimize the number of such separations (see Figure 3a to 3c). In order to do this minimization we can first choose an optimal permutation of the transistors. Furthermore, we can use the commutativity and idempotence of logical-and and logical-or to restructure the series-parallel graph.

3. Graph theoretic formulation of the layout problems

We represent a pulldown network as a graph $G = (V,E)$. The edges represent the transistors. Equipotential regions are represented by vertices. The two vertices an edge joins are the source and drain of the corresponding transistor. We require $G$ to be a series-parallel graph. $|V|$ shows that a series-parallel graph is uniquely defined by its decomposition tree $T$. Here $T$ is a partially ordered tree. Specifically, the children of each series-node are ordered. Subtrees of $T$ represent subgraphs of $G$ that are again series-parallel (see Figure 4). Obviously, a network represented by $G$ can be laid out without separations exactly iff $G$ has a Eulerian path.

We can see from the following example that we must exploit the commutativity of the logical-and to get a minimum width layout. Figure 5a shows a network implementing $(A+B)C + F(D+E)$. Figure 5b shows the corresponding series-parallel graph. This graph isn't Eulerian since it has 4 vertices with odd degree. Figure 6a shows the logically equivalent circuit that results from commuting the terms $F$ and $D+E$ in the second logical-and. The corresponding graph (Figure 6b) is Eulerian. $ABCFDE$ is a Eulerian path.

Definition 1: Two series-parallel graphs $G$ and $H$ are called sp-equivalent (write: $G \sim_{sp} H$) if the decomposition trees of $G$ and $H$ are the same up to orderings of the series-nodes.

[O] gives an algorithm which finds for each series-parallel graph $G$ a sp-equivalent Eulerian graph, if such a graph exists. Figure 7 gives an example of a series-parallel graph $G$ that has no sp-equivalent Eulerian graph. In such a case we have two alternatives for minimizing the width of the layout:

1.) Minimum Number of Duplications Problem (MNDP).

Because of the idempotence of logical-or, we can replace each transistor by a pair of transistors put in parallel and gated with the same signal. This results in duplicating the corresponding edge in the series-parallel graph for the network. We thus choose a minimum number of such duplications to make the graph Eulerian (Figure 8).

2.) Minimum Number of Separations Problem (MNSP).

We look for a sp-equivalent graph that can be edge-covered with a minimum number of edge-disjoint paths. This minimizes the number of separations (Figure 9).

4. Minimum Number of Duplications Problem
4.1 Structural Analysis

Definition 2: A trail in the series-parallel graph $G$ is a path that contains each edge of $G$ at least once.

Problem MNDP: Given a series-parallel graph $G$, find a series-parallel graph $H$ such that $G \sim_{sp} H$ and $H$ has a minimum length trail $w$. In this case $(H,w)$ is the solution of the instance $G$ of MNDP.