Self-Organizing Kohonen Maps for FPGA Placement

David C. Blight and Robert D. McLeod

Department of Electrical and Computer Engineering University Of Manitoba
Winnipeg, Manitoba Canada R3T 2N2

Abstract. In this paper we present a novel placement algorithm for FPGAs. This algorithm is based upon the self-organizing map used in unsupervised learning algorithms for artificial neural networks performing pattern classification. The self-organizing map is used to map the connectivity of the design to a two dimensional regular mesh topology. This is followed by simple compaction to minimize wire costs.

1 Introduction

The popularity of Field Programmable Gate Arrays (FPGA) has been dramatically increasing in recent years. Consisting of a large array of identical programmable Lookup-Tables (LUTs) blocks connected by programmable interconnects, these devices may be programmed to implement both combinational and sequential circuits. Although each FPGA device contains a large number of LUTs, it is not always possible to utilize all of the available resources for a particular design. In order to increase the utilization of available resources it is important to develop tools which can efficiently synthesize design specifications into a programmed device. FPGA tools typically divide this process into three steps. First the design specification is partitioned into simple functions, each of which is implementable by the LUTs in the device. Next each function is assigned to a specific LUT in the FPGA device. Finally the interconnects between functions are routed. In this paper we focus on the placement problem of assigning functional tasks to individual LUTs in the FPGA such that constraints based on area and performance are optimized.

Placement algorithms utilized in CAD tools for FPGAs have traditionally employed algorithms originally developed for other technologies including Printed Circuit Boards (PCB) and Integrated Circuit (IC) layouts. Although the placement problem for FPGA is closely related to those of other technologies, the basic structure of FPGA is different from other technologies. FPGA are implemented as a prefabricated array of programmable logic blocks and interconnects. The regular structure of such devices make the use of alternative placement algorithms attractive. In this paper we present a placement algorithm developed for FPGAs based on the unsupervised learning algorithms used in Artificial Neural Networks (ANN) [2]. The Kohonen self organizing map [1] is used to first map the connectivity of the design to a two dimensional regular mesh topology, and
then simple one and two dimensional compaction algorithms may be used to produce an area efficient and highly routable mapping.

The remainder of this paper is organized as follows. Section 2 introduces the Kohonen self organizing map. Section three presents placement algorithms for FPGAs utilizing the Kohonen map algorithm. The next section presents some results and comparisons of the new algorithm with the commonly used simulated annealing algorithm [3]. The final section discusses the usefulness of this algorithm as well as future work.

2 Kohonen Self Organizing Map

The Kohonen self organizing map is an unsupervised learning algorithm for ANNs initially developed for applications including speech and pattern recognition. Although FPGA placement problem does not appear to be closely related to the pattern recognition problem, the self organizing map produces a spatially ordered map of its input signals. It is this spatial ordering which we will utilize in the placement algorithm.

The Kohonen map exhibits two essential effects which produce the spatial ordering.

- Spatial concentration of neural activity in a neighborhood which best matches the input. This is a product of the competitive nature of the learning algorithm.
- Weight adjustment of the best matching neuron and its topological neighborhood.

2.1 Neural Model

The basic structure of the neural network used in Kohonen maps is show in Figure 1. A two dimensional array of neurons are layed out in a rectangular fashion. Each neuron is connected to the input signals of the network. The input signals can be represented as an n-dimensional vector.

\[ x = [x_0, x_1, \ldots, x_{n-1}] \in \mathbb{R}^n \]  

Each neuron in the network is connected to each of the input signals, and each connection has a weight associated with it. The weights for each neuron \( i \) can also be represented as a vector.

\[ m_i = [m_{i0}, m_{i1}, \ldots, m_{i_{n-1}}] \in \mathbb{R}^n \]

2.2 Learning Algorithm

Learning refers to the determination of weight values on connections in a neural network. Although it may be possible to analytically determine the optimal