INVESTIGATING THE EFFICIENCY OF THE VPR AND COFFE AREA MODELS IN PREDICTING THE LAYOUT AREA OF FPGA LOOKUP TABLES

by

Mousa Al-Qawasmi

B.Eng. Electrical Engineering, McGill University, Montreal, Quebec, Canada,

2015

A thesis submitted to Ryerson University
in partial fulfillment of the requirements
for the degree of

Master of Applied Science
in the Program of
Electrical and Computer Engineering

Ryerson University

Toronto, Ontario, Canada, 2020

AUTHOR'S DECLARATION

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

ABSTRACT

Investigating the Efficiency of the VPR and COFFE Area Models in Predicting the Layout Area of FPGA Lookup Tables

Mousa Al-Qawasmi, Master of Applied Science, 2020
Electrical and Computer Engineering, Ryerson University, Canada

A single tile in a mesh-based FPGA includes both the routing block and the logic block. The area estimate of a tile in an FPGA is used to determine the physical length of an FPGA's routing segments. An estimate of the physical length of the routing segments is needed in order to accurately assess the performance of a proposed FPGA architecture. The VPR (Versatile Place and Route) and the COFFE (Circuit Optimization for FPGA Exploration) tools are widely used mesh-based FPGA exploration environments. These tools map, place, and route benchmark circuits on FPGA architectures. Subsequently, based on area and delay measurements, the best architectural parameters of an FPGA are decided. The area models of the VPR and COFEE tools take only transistor size as input to estimate the area of a circuit. Realistically, the layout area of a circuit depends on both the transistor size and the number of metal layers that are available to route the circuit. This work measures the effect of the number of metal layers that are available for routing on FPGA layout area through a series of carefully laid out 4-LUTs (4-input Lookup Tables). Based on measured results, a correction factor for the COFFE

area equation is determined. The correction factor is a function of both the transistor drive strength and the number of metal layers that are available for routing. Consequently, a new area estimation equation, that is based on the COFFE area model, is determined. The proposed area equation takes into consideration the effect of both the transistor drive strength and the number of metal layers that are available for routing on layout area. The area prediction error of the proposed area equation is significantly less than the area prediction errors of the VPR and COFFE area models.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to Dr. Andy G. Ye for his assistance in the preparation of this manuscript.

DEDICATION

To my beloved parents and siblings.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Overview

FPGAs are attractive design platforms due to their non-recurring engineering costs [1], their post fabrication programmability, and their ability to fully exploit inherent parallelism in many digital designs [2]. FPGAs exhibited great improvements in speed, power, and density since their introduction in 1985 [3]. Today, and as a result of technology and architectural improvements, FPGAs are deployed to implement many complex digital systems and System on Chip (SoC) designs with various embedded functionality. FPGA architecture is a well-studied art. Typically, different FPGA architectural designs can be created by varying the numerous architectural parameters that define the logic blocks [4] [5] [6] [7] [8] and routing architectures [9] [10]. Significant improvements in FPGA performance, power, and area can be realized by fully exploiting design spaces to find the best combination of architectural parameters [11] [12] [13].

The area of a single tile in an FPGA is an important metric for optimization purposes [14]. The area models of the VPR [15] and COFFE [16] tools take only transistor size and transistor count as inputs to estimate the area of an FPGA tile. Due to the simplicity of their area models, the VPR and COFFE tools do not

consider the effect of the number of metal layers on FPGA area. The area models do not provide a user with the number of metal layers used to route a circuit. In this work, we measure the effect of varying the number of metal layers available for routing on the layout area of 16 to 1 encoded multiplexers in 4-input Lookup Tables. Based on measured layout area results, we determine a correction factor for the COFFE area model. The correction factor is a function of both the transistor drive strength and the number of metal layers that are available for routing. Consequently, we create a new set of area equations for layout area estimation. The corrected area estimation equation is based on the COFFE area model; and it takes into consideration the effect of both the transistor drive strength and the number of metal layers that are available for routing.

1.2 Thesis Outline

This thesis is organized as follows:

Chapter 1 presents an introduction, and an overview of this work.

Chapter 2 presents an overview of FPGA architecture where Cluster-Based Logic Blocks (CLBs), Basic Logic Elements (BLE), and Lookup Tables (LUTs) are discussed. Moreover, the chapter discusses the VPR and COFFE area models.

Chapter 3 discusses the methods followed to create the 4-LUT layouts from which a correction factor for the COFFE area model is determined.

Chapter 4 proposes a correction factor for the COFFE area model that factors in the effect of the number of available metal layers on the layout area estimates.

Chapters 5 concludes this work.

Chapter 2

BACKGROUND

In this section we present an overview of LUT-based FPGA architecture. Most SRAM-based FPGAs use LUT-based logic blocks [17]. The LUT-based FPGA architecture is chosen in this work because it is easy to vary the functionality of a logic block by changing the number of inputs to the LUT [4]. A k-input LUT can implement any k to 1 combinational function with its k inputs. Research shows that the LUT input number k is an important factor that provides a trade-off between FPGA area and performance [18].

Li et al. study the impact of LUT size, routing architecture, and the impact of cluster size on the power-delay product performance measure in FPGAs [18]. Their results show that for all routing architectures, and for all cluster sizes, k=4 results in up to 1.5x and 1.9x smaller power-delay product. Additionally, their results show that larger cluster sizes (N=8 or N=12) lead to less power consumption compared to smaller cluster sizes (N=4). Rose et al. study the effect of logic block functionality on area efficiency [4]. Their results show that logic blocks with values of *k* between 3 and 4 achieve the lowest total area [19]. Their results also show that this minimum occurs with very little dependence on the programming technology. Based on these

results, we choose k=4 as the number of inputs for the LUTs that we implement in this work.

2.1 FPGA Architecture

2.1.1 Cluster-Based Logic Blocks (CLBs)

Logic blocks are the primary components that implement logical functionality in an FPGA [14]. A logic block can be built by pairing a k-input Lookup Table (k-LUT) with a flip-flip to form a Basic Logic Element (BLE) [19]. N BLEs are grouped together to form a Cluster-based Logic Block (CLB). The structure of a cluster-based logic block is illustrated in Figure 1. The advantage of CLBs is that they allow the sharing of input and output signals within a cluster. Many additional features have been added to the logic elements in FPGAs. Some of today's logic elements in FPGAs include additional logic to improve arithmetic operations. Some logic block architectures utilize LUTs with fracturable structures so that larger LUTs can be split into multiple smaller LUTs.

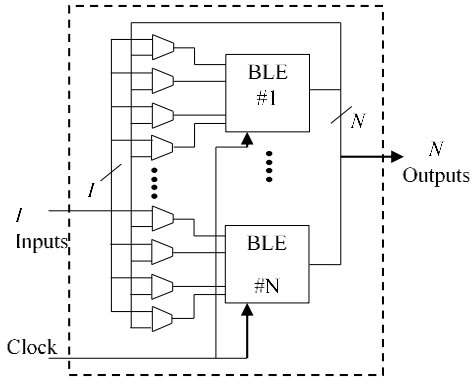

Fig. 1: Cluster-Based Logic Block (CLB)

2.1.2 Basic Logic Elements (BLEs)

In LUT-based FPGA architecture, a basic logic element (BLE) consists of a k-input lookup table (k-LUT), and one D-type flip-flop (DFF) [19]. The k-LUT enables the implementation of any k to 1 combinational logic function. The DFF enables the implementation of sequential logic, which is a fundamental component of digital circuits. Figure 2 illustrates a BLE in a LUT-based FPGA architecture. As depicted in the figure, the output of the k-input LUT can be either registered or unregistered.

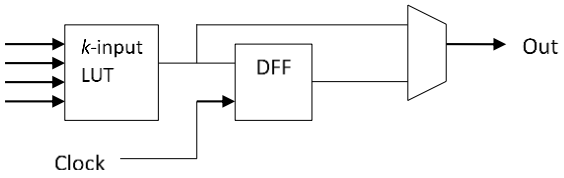

Fig. 2: Basic Logic Element (BLE)

2.1.3 Lookup Tables (LUTs)

The advantage of LUTs is that they provide a high level of functionality. A *k*-input lookup table can implement any function with *k* inputs. A *k*-input LUT can implement 2^k functions. The disadvantage of LUTs is that they become very large for more than five inputs. This is because the number of SRAM cells needed for the *k*-LUT is 2^k. Larger LUTs are often largely underutilized. In this work we study 4-input LUTs. Research shows that k=4 is the optimal number of inputs for LUTs in terms of power-delay product and FPGA area-efficiency [18].

The schematic of a 4-LUT is illustrated in Figure 3. As depicted in the figure, a 4-LUT is mainly comprised of buffers, inverters, SRAM cells, and a 16 to 1 pass-transistors based multiplexer tree.

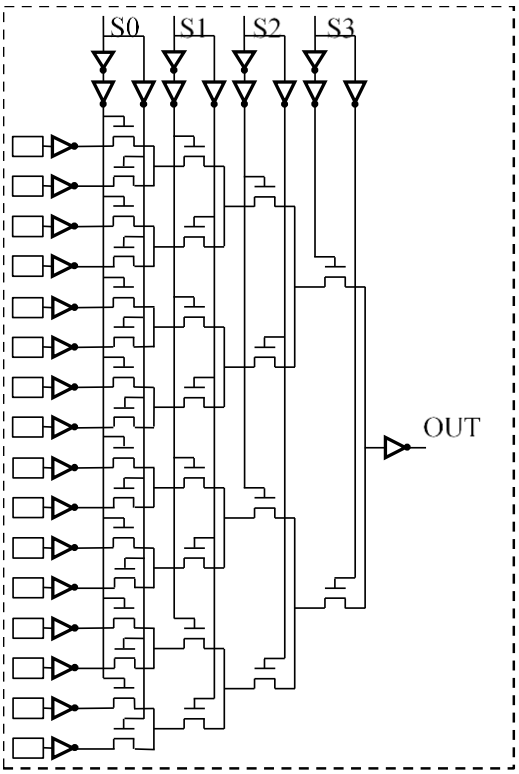

Fig. 3: Schematic of a 4-LUT.

The SRAM cells provide the configuration signals for the LUT. SRAM cells are required to read, write, and hold data. One disadvantage of SRAM cells is that they are volatile – SRAM cells need power to maintain memory. While normal flip-flops can accomplish the requirement of reading, writing, and holding data; SRAM cells are preferred because they are an order of magnitude smaller than flip-flops [20]. Moreover, the small cell sizes of SRAMs offer shorter wire lengths, and hence

lower dynamic power consumption [20]. This is a good trade-off in the case of *k*-LUTs where the memory cells dominate the area.

Multiplexers are key components in constructing LUTs as well as FPGA routing resources. A multiplexer selects an output from several inputs based on a select signal. Due to the presence of the multiplexer, the ON path of the LUT carries binary information in the form of voltage or current to the output of the LUT. As illustrated in the schematic of the 4-LUT shown in Figure 3, the 16 to 1 multiplexer is built using multiple, pass-transistor based, 2 to 1 multiplexers. In pass-transistor circuits, inputs are applied to the source or drain diffusion terminals as opposed to only the gate terminals. Typically, pass-transistor circuits have significant area, speed, and power advantages over static CMOS circuits [20].

2.2 VPR and COFFE Area Models

The most accurate way to determine the area of an FPGA is to create a complete layout [16]. However, such an approach is impractical due to the iterative nature of the FPGA design process. A widely used technique to estimate the area of an FPGA is through the minimum width transistor area model [21]. This section discusses the original minimum width transistor area model used in the VPR tool, and a new version of the minimum width transistor area model that is used in the COFFE tool.

2.2.1 VPR Area Model

The area of one minimum width transistor (size 1x) is defined as the size of the smallest possible contactable transistor in a process plus its spacing to neighboring cells as illustrated in Figure 4. Under this formulation, the model estimates the area of one minimum width transistor with drive-strength x using the equation given in (1).

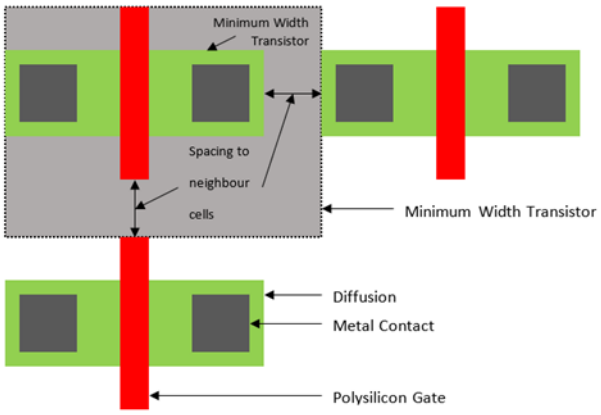

Fig. 4: Minimum-Width Transistor Area Model

$$Area(x) = 0.5 + 0.5x \quad (1)$$

The minimum width transistor area model used in the original VPR tool estimates the area of a transistor with drive-strength *x*, in units of minimum width transistor areas, using the equation given in (1). To estimate the area of an FPGA sub-circuit, the model sums up the areas of all the transistors in the sub-circuit [16]. Research shows that this model over-predicts transistor area by up to 143% when compared to area measurements extracted from manual layouts [16]. The developers of the model suggest that there is room for future work to improve on the accuracy of the area model to capture more realistic layout practices [21].

2.2.2 COFFE Area Model

COFFE uses a new, and more accurate, version of the minimum width transistor area model [16]. The model assumes square layouts due to their smaller area. To keep the shape of the layouts square when the drive-strength of transistors is increased, the model combines both diffusion widening and parallel diffusion regions (folding). Another way COFFE improves on the accuracy of the minimum width transistor area model is by using a least-square fit of 65nm layout areas versus transistor drive-strength to obtain area as a function of drive-strength. Other enhancements in the model account for well-sharing and N-well spacing.

The area of NMOS pass-transistors is calculated by COFFE using (2). The area of CMOS transistors is calculated by COFFE using (3).

$$Area(x) = 0.447 + 0.128x + 0.381\sqrt{x} \quad (2)$$

$$Area(x) = 0.518 + 0.127x + 0.428\sqrt{x} \quad (3)$$

The number of metal layers available to route a circuit is an important factor that affects the layout area of the circuit. Both the original and the newer version of the minimum width transistor area model used in VPR and COFFE respectively do not account for the effect of the number of metal layers available to route a circuit on its layout area. This work develops a correction factor for the more accurate COFFE area model that accounts for the effect of the number of metal layers available for routing a circuit on its layout area.

Chapter 3

METHODOLOGY

To obtain the correction factor for the COFFE area model, we normalize the measured area results of our best-effort layouts of 16 to 1 encoded multiplexers against the COFFE predicted area. We layout the 16 to 1 encoded multiplexers using transistors sized at 1x, 6x, and 16x. We route the multiplexers using 2, 3, 5, and 7 metal layers; and we record the actual layout area of the multiplexers. Then, we use the MATLAB curve-fitting tool to curve-fit the COFFE normalized layout area results to a generalized polynomial that is function both the transistor drive strength, and the number of metal layers that are available to route the multiplexers. Based on the results of the curve-fit, a correction factor for the COFFE area model is determined.

We follow the following conventions to layout the multiplexers:

- We use the generic CMOS 8-metal technology file and the Magic VLSI software [22] to implement all the layouts. Note that the technology file reserves the last metal layer (metal layer 8) exclusively for power and clock routing.
- The transistors in the layouts are placed as close as possible to one another without violating the process's design rules.

- A non-preferred routing approach is followed to route the multiplexers. In this routing approach, the direction of the routing is independent of the metal layer used. This approach for routing is followed in order to avoid layout area penalties that may occur due to the introduction of additional routing channels.

Under the prior mentioned formulation, and since larger multiplexers can be constructed from multiple 2 to 1 multiplexers; we start by laying out a simple 2 to 1 multiplexer. Subsequently, we explore the design space of 4 to 1 encoded multiplexers using combinations of multiple 2 to 1 multiplexers. Based on the results of this exploration, we propose the best-effort layouts of 16 to 1 encoded multiplexers for 4-LUTs implemented using transistors sized at 1x, 6x, and 16x, and routed using 2, 3, 5, and 7 metal layers. Note that the layout area data for the 16 to 1 encoded multiplexers sized at 1x, 6x, and 16x, where each of these layouts is routed using 2 metal layers and 3 metal layers, is extracted from other work [23] [24].

3.1 2 to 1 Encoded Multiplexer

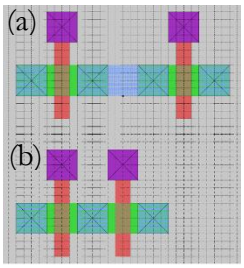

Fig. 5: Layout View of 2 to 1 Multiplexer (a) Without Diffusion Sharing. (b) With Diffusion Sharing

Figure 5 (a) illustrates the layout view of a 2 to 1 multiplexer without diffusion sharing. Figure 5 (b) illustrates the layout view of a 2 to 1 multiplexer with diffusion sharing. Notice in Figure 5 (b) that the source diffusion nodes of the two nMOS transistors are shared. In good layouts, diffusion nodes are shared whenever possible to reduce the diffusion capacitance and to reduce the layout area of a circuit. Reducing the diffusion capacitance reduces the parasitic delay of a circuit and results in better overall performance.

The layout area of the 2 to 1 multiplexer in Figure 5 (a) without diffusion sharing is 392 λ^2. Here, 2λ is the minimum feature size and the transistor length. The layout area of the 2 to 1 multiplexer in Figure 5 (b) with diffusion sharing is 280λ^2. The technique of diffusion sharing reduces the layout area of a 2 to 1 multiplexer by

40%. Based on these results, we build the larger multiplexers using the 2 to 1 multiplexer in Figure 5 (b).

3.2 4 to 1 Encoded Multiplexers

In this section we explore the design space of the 4 to 1 encoded multiplexer for a 2-LUT using combinations of three 2 to 1 multiplexer devices shown in Figure 5 (b). Each 2 to 1 multiplexer is made up of a line of two nMOS transistors where the source diffusion nodes of the transistors are shared. Consequently, the placement problem for the layout of a 2-LUT can be reduced to the most efficient way of placing three straight lines, where each line can be rotated 90^{o} relative to the other lines.

Under this problem formulation, Figure 6 (a) shows all the possible combinations of placing three horizontally oriented 2 to 1 multiplexer devices to design a 4 to 1 encoded multiplexer for a 2-LUT. Since each of the three lines can have either a horizontal or a vertical orientation, each placement configuration in Figure 6 (a) can have 2^3, a total of 8, different variations. For example, the placement configuration in Figure 6 (a) (ii) can have the 8 placement variations illustrated in Figure 6 (b).

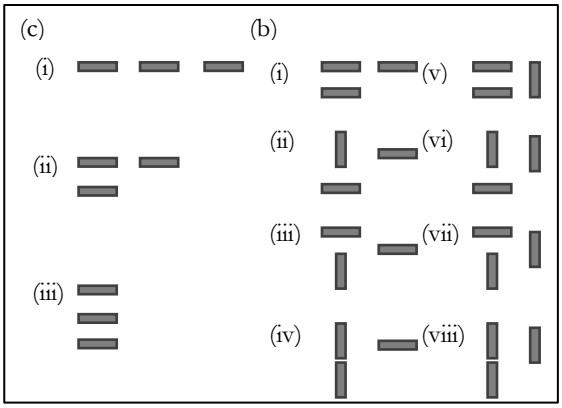

Fig. 6: Different placement configurations of 4 to 1 multiplexers using: (a) Only horizontally oriented 2 to 1 multiplexers. (b) A mixture of horizontally and vertically oriented 2 to 1 multiplexers

Since we are constrained by the DRC rules, we find that the best placement strategies towards laying out area-efficient 4 to 1 encoded multiplexers for 2-LUTs involve providing as much polysilicon sharing (gate sharing) as possible between 2 to 1 multiplexer devices that utilize the same select lines. Hence out of all 24 placement configurations for 4 to 1 encoded multiplexers for 2-LUTs, Figure 7 (a) outlines the final placement configurations that we qualify for the routing process. The layouts of the four placement configurations for 4 to 1 encoded multiplexers after routing are shown in Figure 7 (b). The layout area results for the 4 to 1 encoded multiplexers for 2-LUTs shown in Figure 7 (b) are outlined in Table I. It is clear from the results that the placement configuration illustrated in Figure 7 (b) (v) occupies the least area; however, this placement configuration utilizes a mixture of horizontally and vertically oriented polysilicon gates, which is not compatible in

many processes. Therefore, the placement configurations in Figure 7 (b) (iv) and Figure 7 (b) (v) are disqualified.

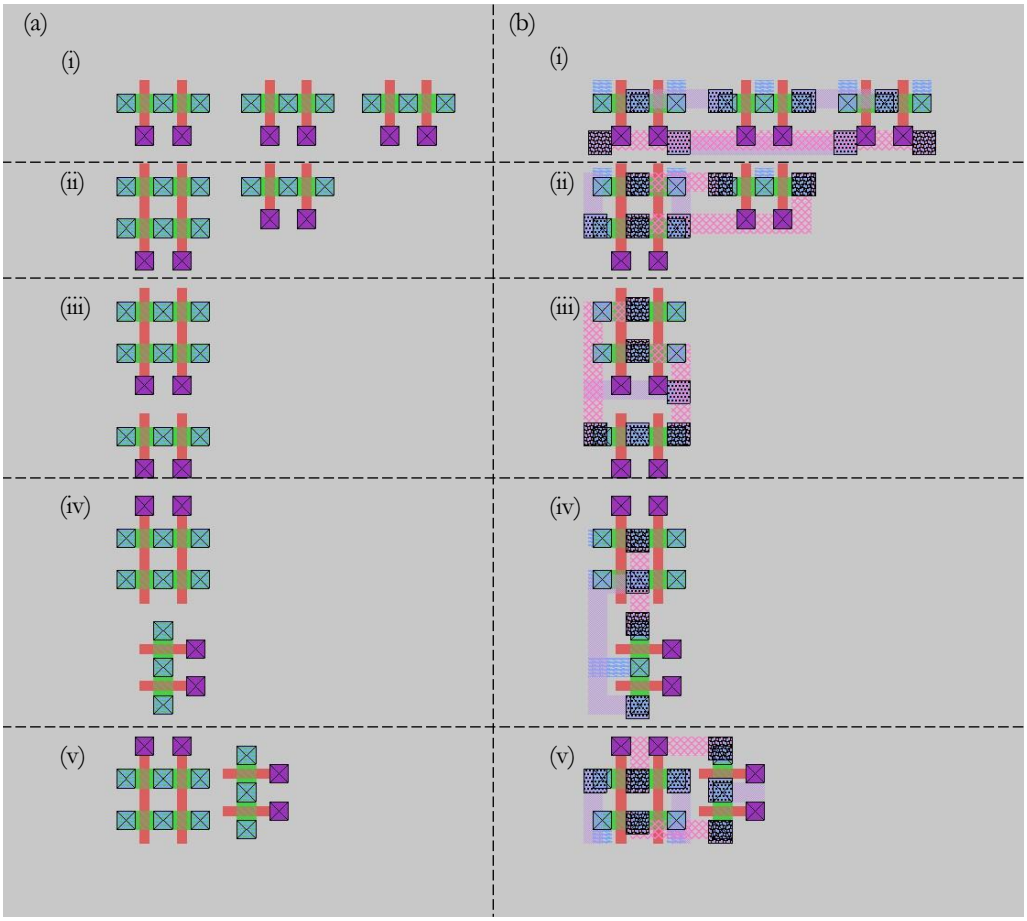

Fig. 7: (a) Different 4 to 1 multiplexer transistor placement configurations, and (b) their corresponding layouts after routing

Table I: Layout Area Results for 4 to 1 Encoded Multiplexer Configurations in Figure 7

Configuration in Figure 7 (b)	Metal Layers	Layout Area	VPR Area	VPR Deviation (%)
(i)	3	1200	1248	-4.0
(ii)	3	1150		-8.5
(iii)	3	943		-32.3
(iv)	3	1008		-23.8
(v)	3	897		-39.1

3.3 16 to 1 Encoded Multiplexers

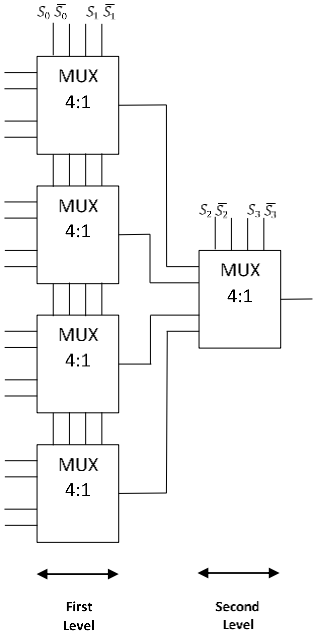

Fig. 8: Schematic of 16 to 1 Encoded Multiplexer for 4-LUTs Structured as 2 Levels of 4 to 1 Encoded Multiplexers

In this section we layout two transistor placement configurations for 16 to 1 encoded multiplexers implemented using transistors sized at 1x, 6x, and 16x based on our exploration of the 4 to 1 encoded multiplexer layout area discussed previously. The first transistor placement configuration implemented using minimum width transistors is illustrated in Figure 9. This transistor placement configuration occupies the least area out of the two when implemented using minimum width transistors (transistors sized at 1x) and routed using 7 metal layers. We use the transistor placement configuration outlined in this layout to implement

16 to 1 encoded multiplexers with transistors sized at 1x and 6x, and routed using 5 metal layers and 7 metal layers when required.

We find that we are required to introduce additional channels for routing that result in area penalties if we try to use the first transistor placement configuration to implement a 16 to 1 encoded multiplexer with transistors sized at 16x. Consequently, we use the transistor placement configuration shown in Figure 10 to implement the 16 to 1 encoded multiplexer with transistors sized at 16x.

The 16 to 1 encoded multiplexer for a 4-LUT can be structured as two levels of 4 to 1 encoded multiplexers as shown in Figure 8. In particular, the first level consists of four 4 to 1 encoded multiplexers. The second level consists of one 4 to 1 encoded multiplexer. Each of the first level 4 to 1 encoded multiplexers takes in four inputs from four different SRAM cells of the 4-LUT and generates one output. The four outputs from the first level 4 to 1 encoded multiplexers are then fed to the second level 4 to 1 multiplexer to generate the final 4-LUT output.

3.3.1 16 to 1 Encoded Multiplexer Layout Configuration I

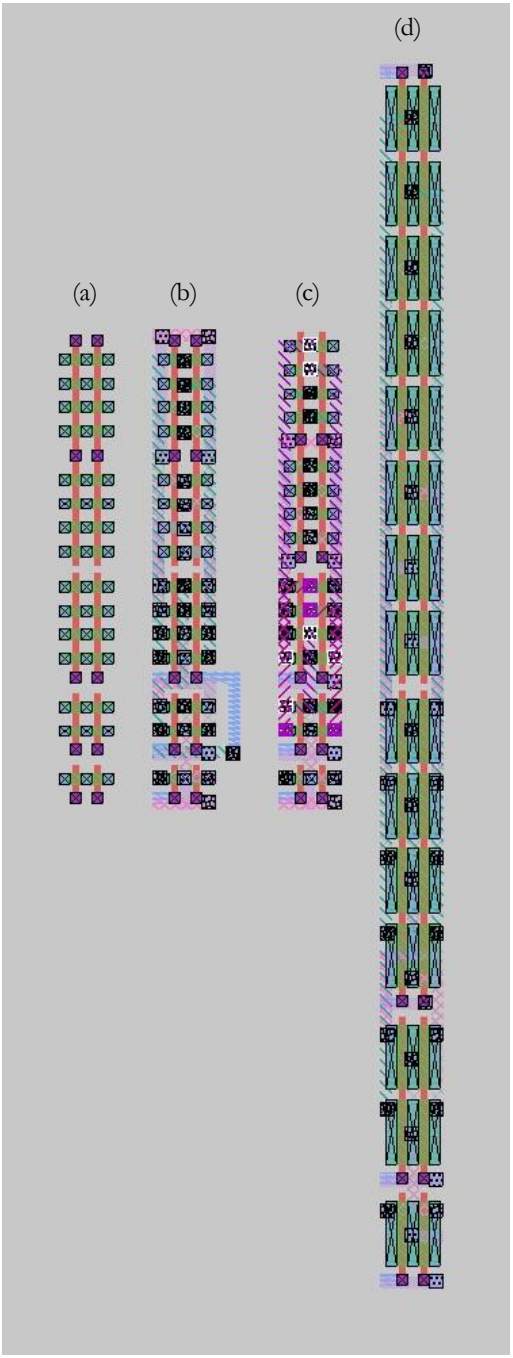

Fig. 9: 16 to 1 encoded multiplexer layout configuration I (a) transistor placement. (b) drive-strength 1x, 5 metal layers. (c) drive-strength 1x, 7 metal layers. (d) drive-strength 6x, 5 metal layers

Given the prior mentioned structure, and since the 4-LUT requires a significant amount of wiring, we build both levels of the 16 to 1 encoded multiplexer with a 4 to 1 encoded multiplexer that utilizes a unidirectional polysilicon gate orientation and occupies the least area for a relatively low amount of wiring. Hence, we design the 16 to 1 encoded multiplexer for a 4-LUT by recursively applying the placement configuration shown in Figure 7 (a) (iii).

Four devices of the 4 to 1 encoded multiplexer shown in Figure 7 (a) (iii) are interleaved with each other to implement the first level multiplexers of the 16 to 1 encoded multiplexer design for a 4-LUT as shown in Figure 9. This method of interleaving allows us to maximize the sharing of polysilicon gates across transistors that share the same control signals. On the downside, this method of interleaving comes at the cost of running long polysilicon lines. Hence, we choose to introduce additional metal wiring and polysilicon contacts to tie down long polysilicon gates. We tie down long polysilicon gates to ensure the uniform distribution of the gate voltages across all transistors that share the same polysilicon lines; and thereby avoid the penalty of degraded performance.

The second level consists of one device of the 4 to 1 encoded multiplexer shown in Figure 7 (a) (iii) that is placed vertically below the first level of 4 to 1 encoded multiplexers as illustrated in Figure 9. Such a placement configuration for the second level 4 to 1 encoded multiplexer is convenient for area-efficiency.

Additionally, the placement of the second level 4 to 1 encoded multiplexer below the first level of 4 to 1 encoded multiplexers allows us to remove any blocking on the path from the SRAM cells of the 4-LUT to the inputs of the 16 to 1 encoded multiplexer that may occur if the second level 4 to 1 encoded multiplexer is placed horizontally to either side of the first level 4 to 1 encoded multiplexers.

The final layouts of the 16 to 1 encoded multiplexer for a 4-LUT using the recursive application of the 4 to 1 encoded multiplexer in Figure 7 (a) (iii) are shown in Figure 9. Although geometrically elongated, this layout occupies the least area out of all the layouts that we implement in the CMOS generic 8-metal process using transistors with drive-strengths of 1x and 6x. The 16 to 1 encoded multiplexer in Figure 9 (c) is designed using transistors with drive-strength of 1x and routed using 7 metal layers. The layout occupies a layout area of 3,520 λ^2. The layout area of the 16 to1 encoded multiplexer in Figure 9 (c) is approximately 44% smaller than the VPR predicted area, and 42% smaller than the COFFE predicted area. One must note that for this transistor placement configuration, the least area is achieved by routing the transistors using 7 metal layers.

We choose to route the 16 to 1 encoded multiplexer implemented using minimum width transistors with 7 metal layers to avoid layout area penalties due to the introduction of additional routing channels. Such area penalties are clear when the layout area of the 16 to 1 encoded multiplexer implemented using minimum width

transistors and routed with 7 metal layers (shown in Figure 9 (c)) is compared to the 16 to 1 encoded multiplexer implemented also using minimum width transistors but routed using 5 metal layers (shown in Figure 9 (b)).

We use the transistor placement configuration illustrated in Figure 9 to implement the layout of the 16 to 1 encoded multiplexer with drive-strength 6x transistors. The 16 to 1 encoded multiplexer with drive-strength 6x transistors is routed using 5 metal layers and is shown in Figure 9 (d). Note that for this layout, no further area reduction can be achieved by using more than 5 metal layers to route the multiplexer.

Since we are constrained by only 7 metal layers in the CMOS generic 8-metal process, we run into routing issues when we try to use the transistor placement configuration illustrated in Figure 9 to implement a 16 to 1 encoded multiplexer using folded drive strength 16x transistors. Consequently, we use a different transistor placement configuration to implement these multiplexers.

3.3.2 16 to 1 Encoded Multiplexer Layout Configuration II

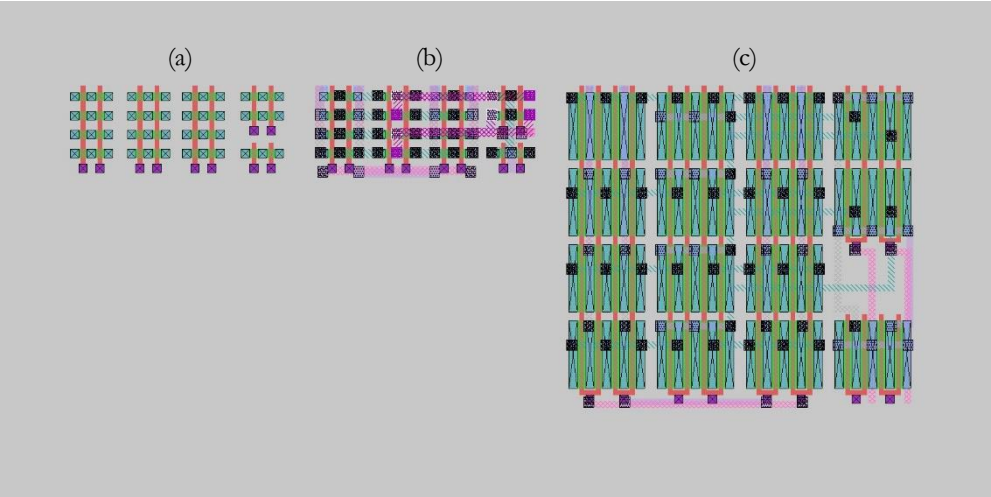

Fig. 10: 16 to 1 encoded multiplexer layout configuration II (a) transistor placement. (b) drive-strength 1x, 7 metal layers. (c) drive-strength 16x, 5 metal layers

The minimum width transistor placement for the second layout is shown in Figure 10 (a), and the final version of the layout after routing is shown in Figure 10 (b). Although this layout occupies a larger area than the earlier-discussed layout shown in Figure 9 (c), this layout facilitates the implementation of designs with larger transistor sizes due to its shorter polysilicon lines and its square-like geometric shape. This layout occupies a layout area of 4,472 λ^2 and uses a total of 7 metal layers. It occupies an area that is 28.3% and 25.8% smaller than the areas predicted by VPR and COFFE respectively. We use this placement configuration to layout the 16 to 1 encoded multiplexer using transistors with drive-strength of 16x and folded once. The 16 to 1 multiplexer implemented using folded drive strength 16x

transistors, and routed using 5 metal layers, is shown in Figure 10 (c). No further area reduction can be achieved with a larger number of metal layers.

Similar to the layout discussed previously in section 3.3.1, we build the 16 to 1 encoded multiplexer using two levels of 4 to 1 encoded multiplexers. The main difference between the placement configurations of both layouts is that in this layout the first level consists of four devices of the 4 to 1 encoded multiplexer shown in Figure 7 (i). To maximize the sharing of control signals (gate sharing), we organize the 4 to 1 encoded multiplexers at the first level in a vertical configuration. To implement the second level 4 to 1 multiplexer, we choose the least area implementation from Figure 7, and we place it horizontally to the side of the first level 4 to 1 encoded multiplexers. Hence, to build the second level of 4 to 1 multiplexer, we choose the 4 to 1 encoded multiplexer placement configuration shown in Figure 7 (iii) and place it to side of the first level 4 to 1 encoded multiplexers as shown in Figure 10.

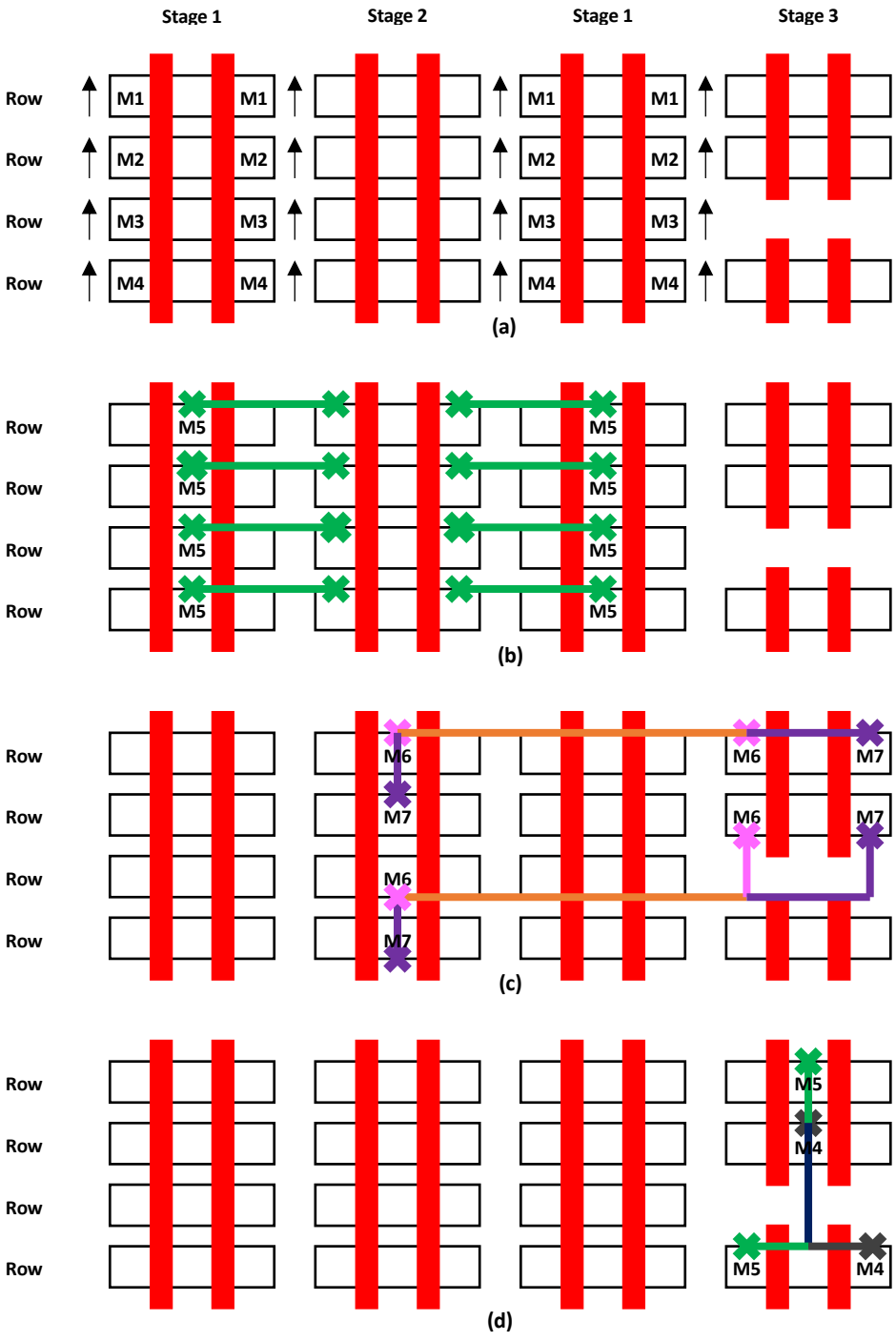

Fig. 11: Illustration of Metal Layer Assignments and Routing Methodology for the 16 to 1 Encoded Multiplexer for 4-LUTs in Figure 10 (c).

The wiring strategy for the 16 to 1 encoded multiplexer shown in Figure 10 (b) is illustrated in Figure 11. Specifically, Figure 11 (a) shows that metal layers assignments for the data inputs of the 16 to 1 encoded multiplexer from the SRAM cells of the 4-LUT. Note that all the inputs of the 16 to 1 encoded multiplexer from the SRAM cells are fed from the top of the layout. To minimize the layout area, these 16 inputs are stacked in 4 tracks with 4 metal layers in each track. Particularly, metal layer 1 is used to connect the first row of transistors that receive SRAM data as inputs to their corresponding SRAM cells. Similarly, the row 2, row 3, and row 4 transistors are connected to their corresponding SRAM cells using metal layers 2, 3, and 4 respectively.

The inputs of the 16 to 1 encoded multiplexer use 4 metal layers to connect their corresponding SRAM cells. Hence, we use metal layer 5 to connect the outputs of the first stage 2 to 1 multiplexers to the inputs of the second stage 2 to 1 multiplexers as shown in Figure 11 (b). Similarly, the outputs of second stage 2 to 1 multiplexers are connected to the inputs of the third stage 2 to 1 multiplexers using metal layers 6 and 7 as shown in Figure 11 (c). Finally, the last stage of 2 to 1 multiplexers, which is also the second level 4 to 1 encoded multiplexer from Figure 7 (iii), is connected internally using metal layers 4 and 5 as illustrated in Figure 11 (d).

We use the transistor placement configuration in Figure 10 (a) and a similar routing strategy to implement 16 to 1 encoded multiplexers using transistors sized at 16x and folded once. The final layout of the 16 to 1 encoded multiplexer for 4-LUTs implemented using transistors sized at 16x (folded once) and routed using 5 metal layers is illustrated in Figure 10 (c).

Chapter 4

EXPERIMENTAL RESULTS

The layout area results for 16 to 1 encoded multiplexers in 4-LUTs implemented using transistors with drive-strengths of 1x, 6x, and 16x and routed using 2, 3, 5, and 7 metal layers are shown in Table II. The VPR and COFFE estimated area estimates are listed in columns 4 and 5 of the table respectively. The percent deviation of the VPR and COFFE area estimates from the area measurements of our manual layouts are listed in columns 6 and 7 of the table respectively.

As shown in the table, the VPR and COFFE percentage deviation (prediction error) from the measured area results that are extracted from the manual layouts change significantly as the number of available metal layers varies. The VPR area prediction error ranges from 52.48% under-prediction to 112.77% over-prediction. Specifically, VPR under-predicts the area of 16 to 1 encoded multiplexers in 4-LUTs implemented using transistors with drive strengths of 16x, and routed using 5 metal layers, by 52.48%. Moreover, VPR over-predicts the area of 16 to 1 encoded multiplexers in 4-LUTs implemented using transistors with drive strengths of 1x, and routed using 2 metal layers, by 112.77%.

The COFFE area model significantly under-predicts/over-predicts layout area for designs with small transistor sizes. The COFFE area prediction error reduces significantly as the transistor size is increased. For example, the COFFE area prediction error ranges from 54.10% under-prediction to 71.25% over-prediction for 16 to 1 encoded multiplexers in 4-LUTs implemented using transistors with drive strengths of 1x. For designs implemented using transistors with drive strengths of 16x, the prediction error reduces to 46.29% under-prediction to 1.60% over-prediction. Based on these results, we propose a layout area equation that is based on the COFFE area model and factors in the effect of the number of available metal layers on the predicted layout area. Our proposed layout area equation is given in equation (4):

$$Area(x, y) = COFFE\ Area(x) * COFFE\ Correction\ Factor\ (x, y) \quad (4)$$

Where x is the transistor drive strength, y is the number of available metal layers, COFFE Area(x) is the COFFE area estimate given in equation (2), and COFFE Correction Factor (x, y) is the COFFE Area Correction Factor given in equation (6).

To obtain the COFFE correction factor we plot the COFFE normalized layout area versus transistor size and the number of available metal layers as shown in Figure 12. The COFFE normalized layout area is calculated using equation (5). The

COFFE correction factor $f(x, y)$, where x is the transistor drive strength and y is the number of available metal layers is generated by the MATLAB curve-fitting tool and given in equation (6).

$$COFFE\ Normalized\ Layout\ Area = \frac{Layout\ Area}{COFFE\ Area} \tag{5}$$

Using equation (4), we calculate the corrected area estimate. The proposed corrected area estimate considers the effect of both the transistor drive strength and the number of available metal layers on the predicted area. The results are shown in Table II.

Table II: Comparison of VPR, COFFE, and Corrected Area Equation Area Prediction-Errors

Transistor Size	Metal Layers	Layout Area (λ^2)	VPR Area (λ^2)	COFFE Area (λ^2)	Corrected Area (λ^2)	VPR Deviation (%)	COFFE Deviation (%)	Corrected Area Deviation (%)
1x	2	13132	6240	6028	12081	52.48	54.10	8.00
	3	11904			11247	47.58	49.36	5.52
	5	5728			5182	-8.94	-5.24	9.53
	7	3520			3682	-77.27	-71.25	-4.59
6x	2	22834	21840	13541	26165	4.35	40.70	-14.59
	3	22158			24691	1.44	38.89	-11.43
	5	10511			12074	-107.78	-28.83	-14.87
	7	10511			9990	-107.78	-28.83	4.96
16x	2	47160	53040	25328	45296	-12.47	46.29	3.95
	3	46080			44033	-15.10	45.03	4.44
	5	24928			24204	-112.77	-1.60	2.91
	7	24928			25120	-112.77	-1.60	-0.77

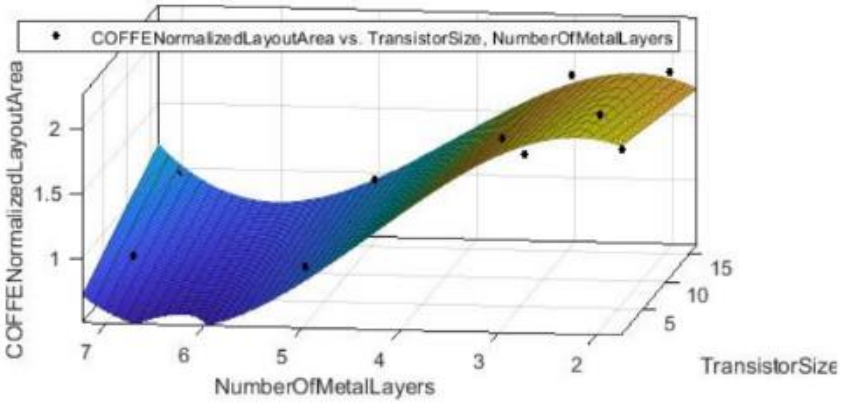

Fig. 12: Plot of COFFE Normalized Area Versus Transistor Size and Number of Available Metal Layers.

Linear model Poly13: (6)

f(x,y) = p00 + p10*x + p01*y + p11*x*y + p02*y^2 + p12*x*y^2 + p03*y^3

Coefficients (with 95% confidence bounds):

p00 = 0.277 (-3.185, 3.739)

p10 = -0.0231 (-0.1753, 0.1291)

p01 = 1.807 (-0.8753, 4.49)

p11 = 0.003326 (-0.0748, 0.08146)

p02 = -0.5546 (-1.185, 0.07605)

p12 = 0.0005147 (-0.008039, 0.009068)

p03 = 0.04325 (-0.002814, 0.08931)

Goodness of fit:

SSE: 0.1743

R-square: 0.9491

Adjusted R-square: 0.8881

RMSE: 0.1867

Notice in Table II that the percentage deviation (prediction error) of our corrected area estimation equation (from actual layout area measurements) is significantly less than the respective VPR and COFFE prediction errors. For 16 to 1 encoded multiplexers implemented using transistors with drive strengths of 1x, the corrected area estimate prediction error ranges from 9.53% under-prediction (5 metal layers) to 4.59% over-prediction (7 metal layers). For designs implemented using transistors with drive strengths of 6x, the corrected area prediction error ranges from 4.96% under-prediction (7 metal layers) to 14.87% over-prediction. For designs implemented using transistors with drive strengths of 16x, the corrected area prediction error ranges from 4.44% under-prediction (3 metal layers) to 0.77% over-prediction (7 metal layers).

Chapter 5

CONCLUSION

We observe that the layout area of the 16 to 1 encoded multiplexers in 4-LUTs that we investigate in this paper significantly changes when the number of available metal layers varies. In particular, for 16 to 1 encoded multiplexers implemented using minimum width transistors, the layout area changes from 13,132 λ^2 to 3,520 λ^2 when the number of metal layers is increased from 2 metal layers to 7 metal layers. These numbers are significantly different from the 6,240 λ^2 and 6,080 λ^2 layout area results predicted by the VPR and COFFE models respectively.

Similarly, for 16 to 1 encoded multiplexers implemented using transistors sized at 6x, the layout area ranges from 22,834 λ^2 to 10,511 λ^2 when the number of metal layers is increased from 2 metal layers to 5 metal layers. These areas also significantly differ from the 21,840 λ^2 and 13,541 λ^2 layout area results predicted by VPR and COFFEE respectively.

For 16 to 1 encoded multiplexers implemented using transistors sized at 16x and folded once, the layout area ranges from 47,160 λ^2 to 24,928 λ^2. These layout area

results also differ from the 53,040 λ^2 and 25,328 λ^2 layout area results predicted by VPR and COFFE respectively.

Based on our measured area results, we propose a layout area equation that is based on the COFFE area model that takes into consideration both the transistor drive-strength and number of available metal layers. The prediction error of our proposed area equation is significantly less than the prediction errors of the VPR and COFFE area models.

As a step towards increasing the accuracy of future FPGA exploration strategies, we suggest that future work should examine more circuit types to create more accurate metal-layer based correction factors for a wider range of FPGA components.

BIBLIOGRAPHY

[1] I. Kuon, and J. Rose, "Measuring the gap between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 26(2), pp. 203-215, 2007.

[2] D. Hill, and N.-S. Woo, "The benefits of flexibility in lookup table-based FPGAs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 12, no. 2, pp. 349-353, 1993.

[3] H. Gao, Y. Yang, X. Ma, and G. Dong, "Analysis of the effect of LUT size on FPGA area and delay using theoretical derivations," in *Sixth international symposium on quality electronic design (isqed'05)*, San Jose, CA, USA, 2005.

[4] J. Rose, R.J. Francis, D. Lewis, and P. Chow, "Architecture of field-programmable gate arrays: The effect of logic block functionality on area efficiency," *IEEE Journal of Solid-State Circuits,* vol. 25(5), pp. 1217-1225, 1990.

[5] J. Rose, R. J. Francis, P. Chow and D. Lewis, "The effect of logic block complexity on area of programmable gate arrays," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, San Diego, CA, USA, 1989.

[6] A. Marquardt, V. Betz, and J. Rose, "Speed and Area Tradeoffs in Cluster Based FPGA Architectures," *IEEE Transactions on VLSI Systems,* vol. 8, pp. 84-93, Feb. 2000.

[7] E. Ahmed, and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 12, no. 3, pp. 288-298, March 2004.

[8] J. Rose, and I. Kuon, "Background," in *Quantifying and Exploring the Gap Between FPGAs and ASICs*, Springer US, 2010, pp. 5-26.

[9] M.I. Masud, and S.J.E. Wilton, " A New Switch Block for Segmented FPGAs," in *International Workshop on Field Programmable Logic and Applications*, August 1999.

[10] Y. Chang, D. F. Wong, and C.K. Wong, "Universal switch modules for FPGA design," *ACM Transactions on Design Automation of Electronic Systems (TODAES),* vol. 1, no. 1, pp. 80-101, Jan. 1996.

[11] A. El Gamal, and J. L. Kouloheris, "FPGA performance versus cell granularity," in *Proceedings of the IEEE 1991 Custom Integrated Circuits Conference*, San Diego, CA, USA, 1991.

[12] S. Singh, J. Rose, P. Chow, D. Lewis, "The effect of logic block architecture on FPGA performance," *IEEE Journal on Solid-State Circuits,* vol. 27, no. 3, pp. 281-287, Mar. 1992.

[13] A. El Gamal, and J. L. Kouloheris, "PLA-based FPGA Area Versus Cell C+ Granularity," in *1992 Proceedings of the IEEE Custom Integrated Circuits Conference*, Boston, MA, USA, 1992.

[14] J. Rose, and I. Kuon, "Chapter 4 Automated Transistor Sizing for FPGAs," in *Quantifying and Exploring the Gap Between FPGAs and ASICs*, Springer US, 2010, pp. 63-90.

[15] J. Rose, and V. Betz, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," in *Proc. Int'l Workshop FieldProgrammable Logic and Applications*, Berlin, 1997.

[16] V. Betz, and C. Chiasson, "COFFE: Fully-automated transistor sizing for FPGAs," in *International Conference on Field-Programmable Technology (FPT)*, 2013.

[17] V. Betz, and J. Rose, "How much logic should go in an FPGA logic block," *IEEE Design & Test of Computers,* vol. 15, no. 1, pp. 10-15, January 1998.

[18] F. Li, D. Chen, L. He and J. Cong, "Architecture evaluation for power-efficient FPGAs," in *Proceedings of the 2003 ACM/SIGDA eleventh international symposium on field programmable gate arrays*, Monterey, California, USA, 2003.

[19] J. Rose, and V. Betz, "Cluster-based logic blocks for FPGAs: area-efficiency vs. input sharing and size," in *Proceedings of CICC 97 - Custom Integrated Circuits Conference*, Santa Clara, CA, USA, 1997.

[20] N. Weste and D. Harris, "Chapter 12 Array Subsystems," in *CMOS VLSI Design*, Addison-Wesley, 2011, p. 499.

[21] A. Marquardt, V. Betz, and J. Rose, Architecture and CAD for Deep-Submicron FPGAs, Kluwer, 1999.

[22] M. V. L. Tool, 1980.

[23] F. Khan and A. Ye, "An Evaluation on the Accuracy of the Minimum-Width Transistor Area Models in Ranking the Layout Area of FPGA Architectures." *ACM Transactions on Reconfigurable Technology and Systems (TRETS) - Special Section on FCCM 2016 and Regular Papers* 11.1 (2018): Article 8.

[24] F. Khan and A. Ye, "An evaluation on the accuracy of the minimum width transistor area models in ranking the layout area of FPGA architectures." *26th Internation Conference on Field Programmable Logic and Applications (FPL)*. 2016. 1-11.