PLA-based FPGA Area versus Cell Granularity

Jack L. Kouloheris* Abbas El Gamal*

Dept. of Electrical Engineering
Stanford University
Stanford, Ca. 94305-4055
(415) 723-4769
FAX: (415) 723-8473

Abstract

The tradeoff between the area of a PLA-based FPGA and its cell granularity is experimentally investigated. We show that the total FPGA area is smallest for a cell with 8–10 inputs, 3–4 outputs, and 12–13 product terms, a larger granularity than the minimum area lookup table cell. The total area for the minimum area PLA-based designs and lookup table designs are comparable when implemented in the same technology.

1 Introduction

Previous works[1, 2] have examined area tradeoffs in multi-output lookup table cells and found that the 4-input, 1-output cell yields the smallest FPGA area of any K-input, M-output lookup table cell for a wide range of programming technologies and routing pitches. For an FPGA with large interconnect delays, however, a cell with more than 4 inputs may yield better performance, since there will be fewer interconnections between the cells and fewer levels of logic[3]. A lookup table with much more than 4 inputs will be prohibitively large because of its exponential growth in size with the number of inputs.

In [2] we examined the number of product terms used per cell in an FPGA based on a multi-output lookup table cell. We found that, on average, the functions mapped into the lookup tables used considerably fewer product terms than the lookup table capacity (see Figure 1). These results indicate that PLAs may be an area-efficient alternative to lookup tables.

In this paper we investigate the best granularity for a PLA cell (in terms of the number of inputs, K, outputs, M, and product terms, N). We then compare the area of an FPGA based on the best PLA cell to one based on the best lookup table cell. We show that the "optimal" granularity for the PLA cell is larger than for the lookup table cell, and that the total FPGA area for the two are comparable when implemented with the same programming technology.

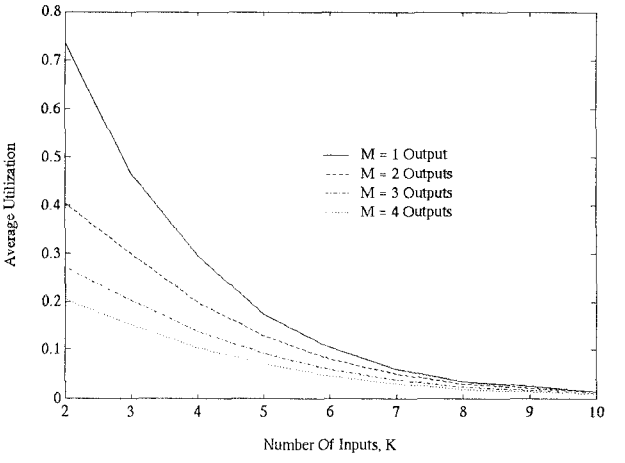

Figure 1: Cell Utilization versus K and M. The cell utilization U, for a K-input, M-output lookup table cell is defined as the ratio of the number of product terms used per cell to the cell capacity, $M \cdot 2^{K-1}$.

2 Multi-Output PLA Cells

2.1 FPGA Model

The FPGA model we assume comprises rows of identical K-input, M-output, N-product term basic cells (denoted (K,M,N)) interspersed with horizontal routing channels. Each cell output is assumed to have an optional D flip-flop in series with it (see Figure 2). We assume the vertical wiring goes over the cell rows. This FPGA model does not directly correspond to any commercial FPGA, but rather tries to capture the essential elements of a PLA-based design.

*Partially supported by DARPA contract J-FBI-89-101

IEEE 1992 CUSTOM INTEGRATED CIRCUITS CONFERENCE 0-7803-0246-X/92 $3.00 © 1992 IEEE

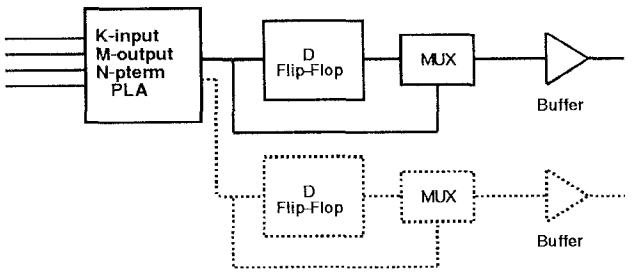

Figure 2: Basic Cell

2.2 Area Model

The PLA cell area model is based on measurements of the layout of a static pseudo-NMOS PLA generated by MPLA[4]. We assume that the elements used to program the PLA are roughly square and have an area A_{pe}. Different programming technologies (e.g. fuses, EPROM cells, etc.) are represented by varying A_{pe}, subject to a minimum size determined by the design rules. The cell width (in lambda) of the layout is given by

$$C_w = \max(18, \sqrt{A_{pe}}) \cdot K + \max(10, \sqrt{A_{pe}}) \cdot M + 98,$$

and the cell height is given by

$$C_h = \max(10, \sqrt{A_{pe}}) \cdot N + 136.$$

Thus the PLA cell area is given by

$$A_{cell} = C_w \cdot C_h + A_{fixed} \cdot M,$$

where A_{fixed} is the area required for the output flip-flops, buffers, and associated fixed internal connections. The routing area per cell, A_{route}, is

$$A_{route} = C_w \cdot T \cdot R_p,$$

where T is the number of horizontal wiring tracks in a wiring channel, and R_p is the routing pitch. Thus the total area for a design of N_{cells} is simply

$$A_{total} = (A_{cell} + A_{route}) \cdot N_{cells}.$$

2.3 Optimizing N

In addition to choosing K and M, we also choose N to minimize the total cell area as follows. Let p_B be the probability mass function of the number of minimized product terms per cell in each benchmark and N_{max} be the maximum number of product terms in any cell in the benchmark. We assume that each lookup table with less than N product terms fits into one PLA and those with more than N product terms must be broken up into several PLAs. For each benchmark and for each value of K and M we want to choose N such that the total cell area, $A_{cells}(N)$, is minimized, where

$$A_{cells}(N) = \sum_{j=1}^{N_{max}} p_B(j) A_{cell} \cdot \left\lceil \frac{j}{N} \right\rceil.$$

A plot of $A_{cells}(N)$ for a typical benchmark is shown in Figure 3.

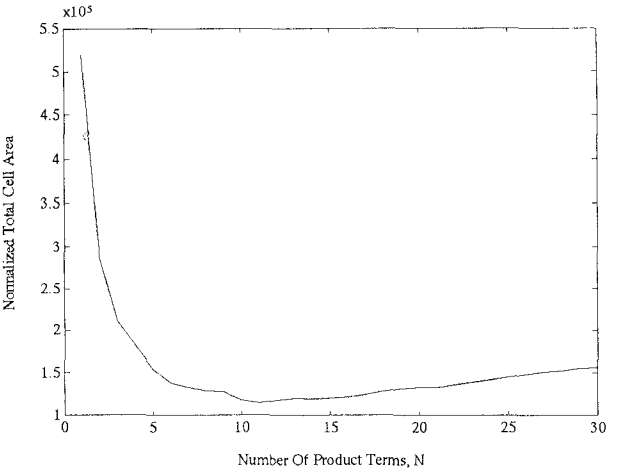

Figure 3: A_{cells} vs. N for a specific benchmark and (K,M)

2.4 Experimental Procedure

We use a set of 20 benchmarks chosen to represent a variety of design types (e.g. random logic, ALU, ECC, etc.) and sources (MCNC[5] and ISCAS benchmark sets, and real FPGA designs) ranging in size from from 80 to almost 9000 2-input cells. Our experimental procedure is as follows:

For each benchmark
  For $K = 2$ to 20
    For $M = 1$ to 4
      1. Partition logic design into K-input equations (Chortle-crf)[6]
      2. Map equations into (K,M,N) cells (DDmap)[1]
      3. Place and Route
      4. Record number of cells and tracks
    End
  End
End

2.5 Area vs. K and M

Figure 4 plots A_{total} versus (K,M) as computed using the area model described in section 2.2[2]. For each

[1] The DDmap algorithm is a modified version of the algorithm presented in [7].

[2] In this figure we have plotted a subset of the benchmarks that does not include the ECC benchmarks (C499 and C1908)

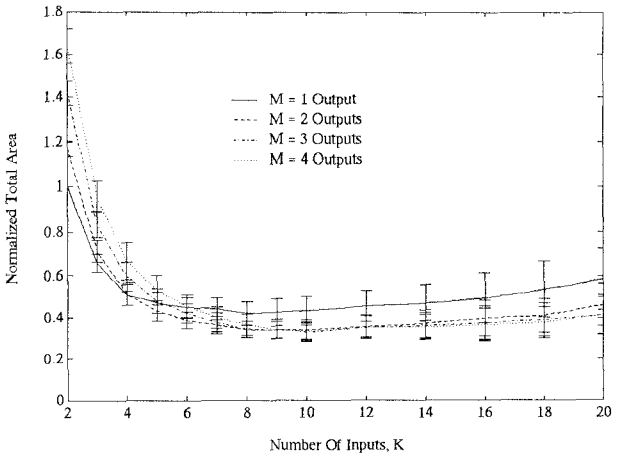

Figure 4: Total Area vs. K and M (The error bars represent a 95% confidence interval of the mean over all benchmarks)

(K,M) and each benchmark, the best N was chosen using the method described in 2.3. For this figure we used a value of $200\lambda^2$ for A_{pe}[9], a value of $13100\lambda^2$ for A_{fixed} and an R_p of 15λ, which roughly corresponds to an EPROM based programming technology. A_{total} is smallest for a PLA with 8–10 inputs, 3–4 outputs, and 12–13 product terms. One output cells are the smallest from K=2 to K=4, two output cells from K=4 to K=7, and three output cells beyond K=7. For K>4, the differences between 2, 3, and 4 output cells are not statistically significant. Varying A_{pe} over a wide range ($200\lambda^2 - 2000\lambda^2$) does not significantly change the best (K,M,N).

2.6 Comparison with Lookup Tables

To compare the relative area efficiency of the lookup table cell and the PLA cell, we compare the smallest lookup table implementation, (4,1), with the smallest PLA implementation, (10,3,12). For a fair comparison, we consider both implementations in the same programming technology, the EPROM cell. We find that A_{total} for the PLA cell implementations ranges from 80% to 300% of the lookup table based implementations. The worst results are obtained with the ECC benchmarks. Exclusive of these two benchmarks, the PLA implementations range from 80% to 130% of A_{total} for the lookup table implementations, and on the average are about the same size. A disadvantage of these PLA-based implementations (as compared to the lookup tables) is that they dissipate static power.

as they distort the trends followed by most of the benchmarks. To handle these sorts of designs well, it would be useful to add XOR gates external to the PLA as done in [10, 11].

On the other hand, the large grained PLA implementations use 25% fewer wiring tracks and 40% fewer levels of logic on average, which would lead to better performance[3].

The PLAs are still relatively sparse, however, and could be improved over the simple-minded model considered here. We collected a number of statistics on the PLAs and found that on average only about half of the inputs are involved in any one product term (Figure 5). This suggests that a fixed product term folding could be used to reduce the size of the PLA. We also found that in the multi-output PLAs, only about 10% of the product terms were shared between outputs (Figure 6). This suggests that fixing the OR-plane (as in PALs) would further reduce the PLA size. On average, less than half of the input variables occur in both true and complement form, which suggests that some of the input inverters could be deleted (Figure 7). Using some or all of these methods would reduce the area for the PLAs to less than that of the lookup tables.

3 Conclusions

We examined the product term utilization of each cell in an FPGA based on (K,M) lookup tables, and found it to be relatively low. This suggested that a PLA might be a reasonable choice for a basic cell, especially for larger granularities. For the PLAs, a cell with 8–10 inputs and 3–4 outputs, and 12–13 product terms yields the smallest total area. When implemented in the same technology as the lookup tables the PLA implementations are about the same size as lookup table implementations, but hold the potential for greater performance because of the larger granularity. Benchmark statistics point the way to reducing the size of the PLAs through folding and other techniques. Future work will involve testing the effectiveness of these techniques as well as investigating other types of cells, such as logic function based cells (e.g. multiplexers, AND-OR gates, etc.).

4 Acknowledgements

We would like to thank Bob Francis for supplying us with the Chortle-CRF[6] technology mapping program. Hubert Hsieh and Harry Hsieh wrote an early version of the DDmap program as a class project under our direction. We would also like to thank the technical staff of Altera and Xilinx for several helpful discussions, Dana How for suggestions on improving the presentation, and Mentor Graphics for the use of their GDT software.

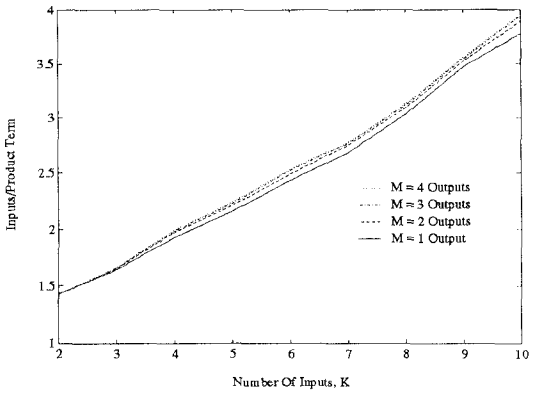

Figure 5: Inputs/Product Term vs. K and M

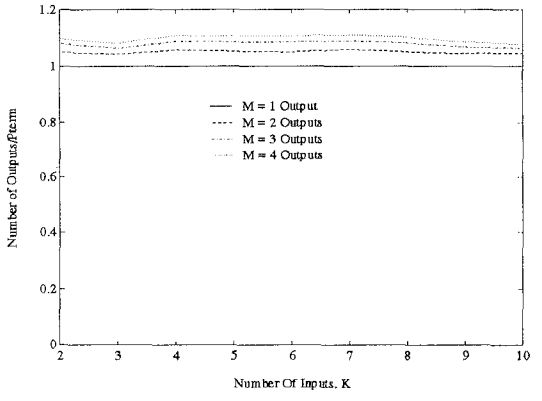

Figure 6: Outputs/Pterm vs. K and M

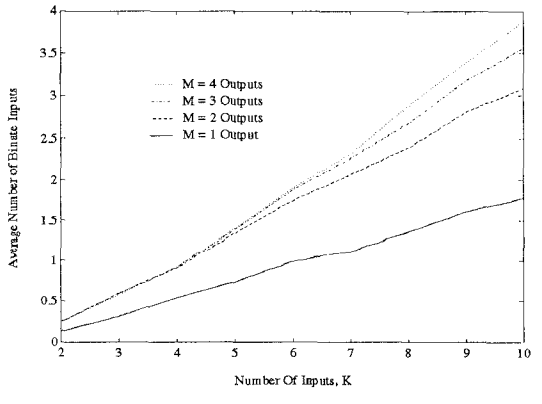

Figure 7: Binate Inputs vs. K and M

References

[1] Jonathan Rose *et al.*, "Architecture of Field Programmable Gate Arrays: The Effect of Logic Block Functionality on Area Efficiency," *IEEE Journal of Solid State Circuits*, Vol. 25, No. 5, October 1990, pp. 1217–1225.

[2] Jack L. Kouloheris and Abbas El Gamal, "FPGA Area vs. Cell Granularity – Lookup Tables and PLA Cells," FPGA '92 Workshop Notes.

[3] Jack L. Kouloheris and Abbas El Gamal, "FPGA Performance vs. Cell Granularity," *Proceedings of the 1991 Custom Integrated Circuits Conference*, pp. 6.2.1–4.

[4] Walter S. Scott *et al.*, editors, "1986 VLSI Tools: Still More Works by the Original Artists," *University of California at Berkeley Report No. UCB/CSD 86/272*, December 1985.

[5] R. Lisanke, "Logic Synthesis and Optimization Benchmarks, User Guide, Version 2.0," *Microelectronics Center of North Carolina*, 1988.

[6] R. J. Francis, Jonathan Rose, Zvonko Vranesic, "Chortle-crf: Fast Technology Mapping for Lookup Table-Based FPGAs," *Proceedings of the 28th ACM/IEEE Design Automation Conference*, June 1991, pp. 227–233.

[7] D. L. Dietmeyer and M. H. Doshi, "Automated PLA Synthesis of the Combinational Logic of a DDL Description," *Design Automation and Fault Tolerant Computing*, Vol III, No. 3/4, 1980, pp. 241–257.

[8] W. Carter *et al.*, "A User Programmable Reconfigurable Gate Array," *Proceedings of the 1986 Custom Integrated Circuits Conference*, pp. 233–235.

[9] R. Hartmann, private communication.

[10] S. C. Wong *et al.*, "A 5000-gate CMOS EPLD with multiple logic and interconnect arrays," *Proceedings of the 1989 Custom Integrated Circuits Conference*, pp. 5.8.1–4.

[11] Cecil H. Kaplinsky, "Programmable Logic Device," U.S. Patent 4,847,612, July 11, 1989.