A Yield Study of VLSI Adders [1]

Zhan Chen and Israel Koren
Department of Electrical and Computer Engineering
University of Massachusetts, Amherst, MA 01003, USA

Abstract

Several 64-bit adders have been designed and their expected yield has been estimated. Our results show that the yield of VLSI adders can be improved by modifying the layout of the original design and/or by choosing a different layout and circuit structure. In certain situations, these approaches can improve the yield by 10% to 17%.

1. Introduction

Designers of VLSI circuits have often two objectives in mind: reduce the area of the circuit and decrease the delay. The purpose of the area reduction is to increase the yield, while the purpose of delay reduction is to improve the performance. During the process of designing a large VLSI circuit, it is common to divide the large circuit into different small functional units, while assigning each unit with pre-specified area and speed requirements. VLSI designers often spend time and make efforts to achieve smaller chip area and higher speed. In many cases however, a smaller area and/or a better performance of a single functional unit will not necessarily result in an area decrease and/or performance improvement for the whole VLSI circuit. As a result, in certain situations, we can choose a design that has a higher yield while still meeting the overall system requirements.

In this paper, we examine this approach by comparing different designs of VLSI adders. We have decided to focus on adders since adders are widely used and therefore, the results reported in this paper can be directly applied to many circuits that comprise adders. Another reason to choose adders is their simplicity compared with other arithmetic units, such as multipliers, floating-point execution units, etc.

There are many ways to design an adder. In this paper, we concentrate on three different kinds of adders: carry-look-ahead adder, carry-skip adder and the hybrid adder used in DEC's ALPHA chip. The main difference between our research and numerous other research works related to VLSI adders is that we focus on yield, while others concentrated on speed.

2. Yield study of adders

2.1 FA standard cell

We started our study with the simplest building block, i.e., the full adder(FA). The MAGIC standard cell library, like any other cell library, contains a layout of a FA. Keeping the whole layout structure unchanged, we used the method introduced in [1] to modify the layout of this standard cell: redistributing the spacing between the conducting lines to reduce the sensitivity of the circuit to short-circuit type defects, and increasing the width of the conducting lines to reduce the sensitivity to open-circuit type defects. The

[1]This work was supported in part by NSF under contract MIP-9305912.

modifications are all in the poly and metal1 layers, and the active areas and the critical path are kept unchanged. SPICE simulations showed that there is no difference in speed between the original and the modified full adders. The yield, as estimated by VLASIC[2], has improved only by about 0.01% compared with the original standard cell. This is a very small improvement, but considering the small area and the full-custom design of the standard cell, this kind of improvement is not insignificant. Suppose we have two circuits of area $1cm \times 1cm$, one of which is composed of an array of the modified full adders, and the other one is composed of the same number of the original full adders. Their yields are 63.72% and 61.79%, respectively. According to today's standard, $1cm^2$ is not a large chip size. Therefore, if the layouts of the standard cells are all modified according to the proposed method, we can expect around 3% yield improvement for a $1cm^2$ chip if the units in the circuit are all designed using standard cells, and the yield improvement rate for each modified standard cell is the same as the full adder. The larger the chip area, the higher the yield improvement. Table 1 shows the relationship between chip area and yield improvement. We may conclude that using the above approach we can still achieve a yield improvement while keeping other parameters unchanged.

	Yield (%)	
Chip area (cm^2)	0.5 × 0.5	1.0 × 1.0
Original full adder	88.94	61.79
Modified full adder	89.25	63.72
Improvement%	0.35	2.79

Table 1: The effect of layout modifications on chip yield of different sizes

2.2 64-bit adders

We have designed three kinds of 64-bit adders, all using 2-micron technology. In each category, we also have some variants. Two yield enhancement techniques have been applied:

Method (1): Redistributing the conducting lines, the same as that used in the FA standard cell design;

Method (2): Moving some wire segements from one layer to another.

Method 2 can improve yield in two ways. First, it can reduce the critical area for short-circuit type defects, since wire segments are moved from a dense region to a sparse region. Secondly, sometimes it can reduce the number of vias, which results in yield improvement. In our examples, we assumed that a via's fault probability is equal to the open-circuit type fault probability of a poly line of length 13 microns and width 2 microns [7], while performance wise, a via is equivalent to a 7.5-micron-long and 3-micron-wide poly line (according to SPICE simulation results).

The adders we have designed include:

(1) Carry-look-ahead adder

We designed two types of carry look-ahead adders:

(i) Ordinary 64-bit carry look-ahead adder

This type of adder has a three-level carry-look-ahead generation with the same blocking factor of 4 at each level[3]. We have designed five versions of this adder: *carry-look-ahead1(a)* is the straight-forward implementation, without taking yield into consideration; *carry-look-ahead1(b)*, which is an improved version of *carry-look-ahead1(a)*, is achieved by modifying the layout of *carry-look-ahead1(a)* using method 1, while keeping its critical path unchanged; *carry-look-ahead1(b')*, another improved version of *carry-look-ahead1(a)*, is similar to *carry-look-ahead1(b)*, but the layout modifications in this adder are restricted to its 4-bit carry-look-ahead building blocks; *carry-look-ahead1(c)* has the same structure as *carry-look-ahead1(a)*, it has however a larger area, and therefore a less dense layout; *carry-look-ahead1(d)* is obtained by modifying the layout of *carry-look-ahead1(a)*, using method 2 in addition to method 1. Partial layouts of *carry-look-ahead1(a)*, *carry-look-ahead1(b)*, and *carry-look-ahead1(d)* are shown in Figure 1.

(ii) Area-efficient 64-bit carry-look-ahead adder

Using the structure presented in [4], *carry-look-ahead2(a)* has four levels of carry-look-ahead, with blocking factors of 2, 4, 4 and 2. *carry-look-ahead2(b)* is the yield enhanced version of *carry-look-ahead2(a)*, using both layout modification techniques.

(2) Carry-skip adder

Following the design reported in [5], this 64-bit adder is divided into four blocks of 16-bit adder each, with an additional block-carry generator. Each 16-bit adder block and the block-carry generator consist of multi-carry-skip paths. The improved version of *carry-skip(b)*, *carry-skip(c)* were obtained from *carry-skip(a)*, in the same way as *carry-look-ahead1(b)*, *carry-look-ahead1(d)* were obtained from *carry-look-ahead1(a)*, respectively.

(3) DEC's ALPHA adder

This is the hybrid adder which was used in the DEC's ALPHA chip and reported in [6]. It combines three types of fast addition techniques: Manchester carry chain, carry-select and conditional-sum[3].

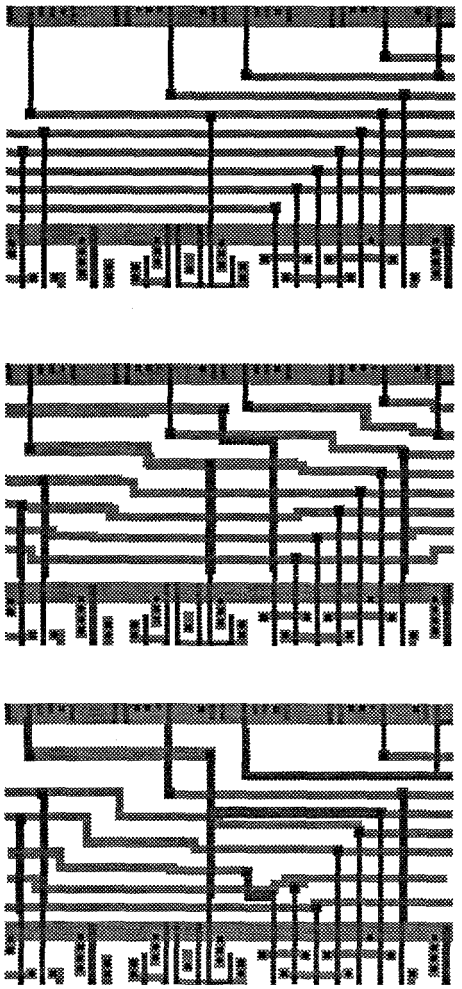
2.3 Results and Analysis

In our study, MAGIC was used to generate the layouts, the performance was analyzed using SPICE, and the yield was calculated using VLASIC. The defect statistics file that we have used was based on the data reported in [7]. The area and delay of the eleven types of adders are shown in Table 2.

Type of adder	Delay (ns)	Area (10^5 μm^2)
carry-look-ahead1(a),(b),(b'),(d)	9.07	9.32
carry-look-ahead1(c)	9.16	10.5
carry-look-ahead2(a),(b)	8.56	8.47
carry-skip (a),(b),(c)	9.92	7.26
ALPHA adder	8.14	7.94

Table 2: Area and speed of different 64-bit adders

Due to the small area of these adders, the difference in yield is small. In order to get a better understanding of their sensitivity to defects, two approaches are used. One is



(a) Original layout

(b) Modified layout using method 1

(c) Modified layout using both methods 1 and 2

Figure 1: Layouts with and without yield enhancement (A part of the data path in *carry-look-ahead1(a)*(a), *carry-look-ahead1(b)*(b), and *carry-look-ahead1(d)*(c)).

duplicating adders in x and y directions to make a larger chip. The results of this approach are shown in Table 3. Another approach is introducing higher defect density, and the results are shown in Figure 2. The higher defect density also has some practical significance, since the first wafers manufactured in a new technology have usually a higher defect density than those manufactured in a mature technology.

Type of adder	Yield of adder arrays (%)					Order	
	1×1	1×2	2×2	3×3	4×4	Yield	Area
carry-look-ahead1(a)	94.67	90.52	81.94	63.88	45.08	11	7,8,9,10
carry-look-ahead1(b)	94.73	90.64	82.19	64.25	45.56	9	7,8,9,10
carry-look-ahead1(b')	94.69	90.57	82.02	63.97	45.24	10	7,8,9,10
carry-look-ahead1(c)	94.81	90.79	82.43	64.73	46.16	8	11
carry-look-ahead1(d)	94.84	90.83	82.52	64.90	46.38	7	7,8,9,10
carry-look-ahead2(a)	95.42	91.96	83.11	68.58	51.14	6	5,6
carry-look-ahead2(b)	95.59	92.28	83.70	69.65	52.51	5	5,6
carry-skip (a)	95.97	93.02	86.54	72.22	56.08	3	1,2,3
carry-skip (b)	96.03	93.14	86.75	72.63	56.63	2	1,2,3
carry-skip (c)	96.11	93.29	87.03	73.15	57.33	1	1,2,3
ALPHA adder	95.85	92.79	86.10	71.41	54.97	4	4

Table 3: Yield of 64-bit adder arrays in different array sizes.

Since the critical paths in the *carry-look-ahead1(b)(b')(d)*, the *carry-look-ahead2(b)* and the *carry-skip(b)(c)* adders are not changed when making layout modifications, they have the same area and speed as *carry-look-ahead1(a)*,*carry-look-ahead2(a)*, and *carry-skip(a)*, respectively. Their yield however, is improved. *Carry-look-ahead1(c)* has even a higher yield than *carry-look-ahead1(b)* due to its less dense layout. *Carry-look-ahead2(a)*, which is an area-efficient adder, has a higher yield, smaller area and better performance than any of the five versions of *carry-look-ahead1*. It seems that its compact layout is responsible for the good results. Compared with *carry-look-ahead1(b)*, the ALPHA adder has 14.8% less area and 1.2% better yield. It is also 10.3% faster than *carry-look-ahead1(b)*. *Carry-skip(c)* has the smallest area and the highest yield among all these adders. But unfortunately, it is also the slowest adder.

From the results shown in Table 3, we can conclude that, in most cases, area and yield have the same order, i.e., the smaller the area, the higher the yield. However, sometimes, a larger area can have a higher yield (for example, *carry-look-ahead1(c)* vs. *carry-look-ahead1(b)*). It is interesting to notice the difference in yield between *carry-look-ahead1(b)* and *carry-look-ahead1(b')*. For a 4×4 adder array, the yield is improved by 1.06% and 0.35% for *carry-look-ahead1(b)* and *carry-look-ahead1(b')*, respectively, compared with *carry-look-ahead1(a)*. This means that yield enhancement due to layout modification, if limited to the sub-cell level, is insufficient. In our examples, the yield improvement rate can nearly double if a system-level yield enhancement effort is made. Therefore, in order to get a better result, system-level yield enhancement should be performed, even if every building block and standard cell have been designed for maximum yield. A similiar result for layer reassignment can also be obtained: the yield improvement rate can be doubled if the method of layer reassignment in addition to conducting line redistribution is used. From Table 3,

we can also find the impact of circuit and layout structures on yield. For example, in the category of carry-look-ahead adders, if only layout modifications are used, the yield can be improved by about 1% (*carry-look-ahead1(b)*) and 2% (*carry-look-ahead1(d)*) for a 4×4 array; but if we choose a different structure, about 16% yield improvement can be achieved (*carry-look-ahead2(b)*).

The relation between defect density and the yield of adders is shown in Figure 2. Here D_0 is the vector of defect density that we used to caculate the yield. It has the value of $\langle 1.65, 0.46, 0.87, 1.31, 3.80, 0.31, 3.79, 2.90, 1.28, 0.66, 0.66, 0.66 \rangle$ per cm^2, corresponding to the defect types of metal1 short, metal1 open, poly short, poly open, metal2 short, metal2 open, poly-metal1 contact open, metal1-metal2 contact open, metal1-active contact open, poly-metal1 pinhole, gate oxide pinhole, and metal1-metal2 pinhole, respectively[7]. As expected, the yield difference of adders increases as the defect density increases. For example, *carry-look-ahead2(a)* and *carry-look-ahead1(a)* have a yield difference of 0.8% when the defect density is D_0, while their difference increases to 9% when the defect density is $10D_0$. This is to say that in a high defect density environment, or during the process of manufacturing a new product, efforts in yield enhancement can be more effective. In our examples, the yield of adders under the environment of high defect density can be improved by about 1% to 3% if only layout modifications are applied (*carry-look-ahead1(a)* vs. *carry-look-ahead1(b)*); however, 5% to 7% yield improvement can be expected if a different layout structure is used (*carry-look-ahead1(a)* vs. *carry-look-ahead2(a)*), while the yield difference between different kinds of adders can be as high as 17% (*carry-skip(c)* vs. *carry-look-ahead1(a)*).

Static CMOS versions of *carry-look-ahead1* and *carry-look-ahead2* have also been designed, and their yield simulations give similiar results as their dynamic versions. Consequently, we do not present these results for the sake of brevity.

3. Conclusion

In this paper, we have studied yield and performance of different adders. Our resulsts show that when choosing a design alternative, we need to take the yield into consideration. The fastest design is not always the best choice. For examples, if we need a 10ns 64-bit adder, then *carry-skip(c)* is a better choice than the ALPHA adder, since it has a smaller area, higher yield and its speed can meet the 10ns requirement; however, if we need a 9ns adder, the ALPHA adder seems to be the best choice.

While choosing a better structure and layout is important for a good yield, the yield of adders can always be improved by modifying the original layout. In certain situations, this approach can improve the yield by 10% to 17%. The layout modification methods introduced in this paper will have no negative effect on speed if we keep the critical path unchanged (for example *carry-look-ahead1(b),(d)* and *carry-skip(b),(c)*). Sometimes, we can even achieve yield enhancement at the cost of area and speed (for example *carry-look-ahead1(c)*), provided that the area and speed requirements of the complete IC can still be met. Based on our experience with the design of these VLSI adders, we conclude that the yield of adders is mainly decided by the layout details of the circuits, that it to say, an adder with a complex layout is more likely to fail than an adder with a simple layout, no matter what specific algorithm they are using. Though these conclusions are based on the study of adders, we can expect them to hold for other logic circuits as well.

References

[1] V.K.R. Chiluvuri and I. Koren, "New Routing and Compaction Strategies for Yield Enhancement," *IEEE Int. Workshop on Defect and Fault Tolerance in VLSI Systems*, pp. 325-334, November 1992.

[2] H. Walker and S. W. Director, "VLASIC: A Catastrophic Fault Yield Simulator for Integrated Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-5(4), pp. 541-556, October 1986.

[3] I. Koren, *Computer Arithmetic Algorithms*, Prentice-Hall, 1993.

[4] T.F. Ngai, M.J. Irwin and S. Rawat, "Regular, Area-time Efficient Carry -look-ahead Adders," *Journal of Parallel and Distributed Computing*, 3(1986), pp. 92-105, March 1986.

[5] T. Sato et al., "An 8.5-ns 112-b Transmission Gate Adder with a Conflict- Free Bypass Circuit," *IEEE Journal of Solid-state Circuits*, 27(4), pp.657-659, April 1992.

[6] D.W. Dobberpuhl et al.,"A 200-MHz 64-b Dual-Issue CMOS Microprocessor," *IEEE Journal of Solid-State Circuits*, 27(11), pp.1555-1565, November 1992.

[7] R.S. Collica et al., "A Yield Enhancement Methodology for Custom VLSI Manufacturing," *Digital Technical Journal*, 4(2), pp.83-99, Spring 1992.

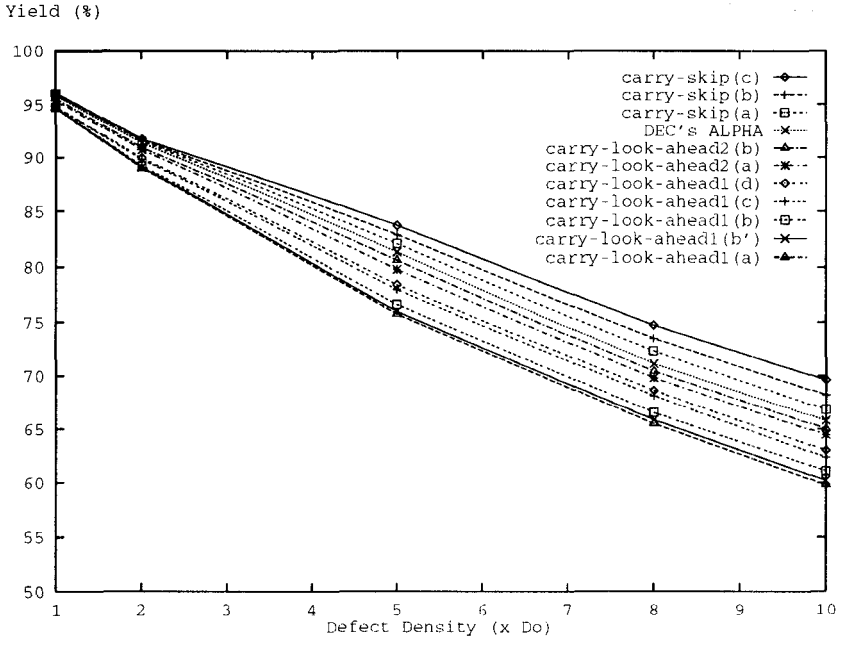


Figure 2: Yield of adders as a function of defect density.