Extreme Area-Time Tradeoffs in VLSI

BINAY SUGLA, MEMBER, IEEE, AND DAVID A. CARLSON, MEMBER, IEEE

Abstract—**Previous research in VLSI has identified the existence of tradeoffs between asymptotic area and asymptotic time for several problems. The existence of a tradeoff for which a constant factor reduction in time could result in a drastic increase in area has been unknown. In this paper, we consider the layout of bounded fan-in and fan-out prefix computation graphs in VLSI and show that the area requirements of such graphs exhibit this interesting property. A small constant factor reduction in time of computation from 2 log η to log η increases the area required to embed an η node prefix computation graph significantly from $O(\eta \log \eta)$ to $\Omega(\eta^2)$. For time of computation T, log $\eta \leq T \leq 2 \log \eta$, the area requirements are $\Omega(\eta^2 2^{2(\log \eta - T)} + \eta)$. We call this behavior an example of an *extreme* area-time tradeoff in VLSI. Since prefix computation also models the carry computation in a carry look-ahead adder, the same behavior is observed in the area requirements of a near-minimum computation time carry look-ahead adder. Finally, we also present circuits which meet the derived lower bounds for all values of T between log η and 2 log η.**

Index Terms—**Area-time tradeoffs, binary addition, lower bounds, minimum bisection width, parallel circuits, prefix circuits, VLSI design.**

I. Introduction

IT IS common knowledge that current VLSI technology is capable of putting a large number of components on a single chip. As a result of this, many of the tools of complexity theory become applicable to VLSI. This in turn has spawned a large number of interesting results in the area of VLSI theory.

Starting with the pioneering work of Thompson [19], the typical approach is as follows. First for a given problem of size η, a lower bound on the requirements of resources—area and time—is established. The resources area and time are usually included in the performance metric (area) $\times$ (time)$^{2\alpha}$, $0 \leq \alpha \leq 1$, where the choice of α determines the relative importance of area and time. Such lower bounds have been obtained for several problems, for example, discrete Fourier transform [19], matrix multiplication [16], binary addition [7], and others [1], [4], [11], [12], [21].

Once the lower bound on the chosen performance metric is known, the designer attempts to devise an algorithm and a corresponding layout which is optimal for a range of values of area and time. Even though a design might be optimal for a certain range of values of area and time, it is nevertheless of some interest to obtain a design for minimum values of time. Recently, for example, Melhorn and Preparata [14] gave an area-time optimal VLSI integer multiplier with minimum computation time.

In this paper, we provide lower and upper bounds for the problem of bounded fan-in, fan-out prefix computation, which also models several naturally occurring computations including *the carry computation in a binary adder,* evaluation of linear recurrences, etc. Through our analysis we present an interesting fact. It is possible for certain problems to exhibit an *extreme* area-time tradeoff in VLSI. More specifically, we show that for the problem of prefix computation, a small constant factor reduction in the time of computation from $2\log\eta$ to $\log\eta$ increases the area required to embed the computation graph in a planar fashion significantly from $O(\eta \log \eta)$ to $\Omega(\eta^2)$.

This result has another ramification for the design of VLSI adder circuits having minimum computation time. Johnson [7] has provided a lower bound of $AT^\alpha = \Omega(\eta \log^\alpha \eta)$ for a VLSI adder. Recently, Brent and Kung [2] provided a regular layout for a parallel adder which was optimal in area for $T = O(\log \eta)$. Since the carry computation in a binary adder can be modeled by prefix computation, our results imply that while the computation graph of a VLSI adder of time 2 log η can be laid out in area η log η, a VLSI adder of time log η (the minimum possible delay time in VLSI) requires $\Omega(\eta^2)$ area.

Our lower bound technique relies on the concept of *minimum bisection width* (MBW) as used by Thompson [19] and others. We derive the lower bound by determining the MBW of all possible computation graphs which can compute prefixes in a certain time T. This is a direct consequence of our ability to exploit the additional structural information arising from the use of bounded fan-in and bounded fan-out gates.

We have organized this paper as follows. First we describe the problem of prefix computation and the VLSI model of computation used here. Then we give the derivation of the lower bound followed by a discussion of upper bounds and area-efficient circuit constructions. Finally, we summarize the results presented here and indicate directions for future research.

II. The Problem

A parallel prefix circuit is a combinational circuit which takes η inputs $x_1, x_2, \cdots, x_{\eta-1}, x_\eta$ and produces the η outputs $x_1, x_1 \circ x_2, \cdots, x_1 \circ \cdots \circ x_\eta$, where $\circ$ is an arbitrary associative binary operation. Parallel prefix circuits have found many applications in the fast addition of two binary numbers [3], [15], regular layouts for parallel adders in VLSI [2], generating consecutive terms of a linear recurrence, simultaneous evaluations of a polynomial at consecutive points of a lattice,

Manuscript received June 8, 1987; revised February 8, 1988.

B. Sugla is with AT&T Bell Laboratories, Holmdel, NJ 07733.

D. A. Carlson is with the Supercomputing Research Center, Bowie, MD 20715.

IEEE Log Number 8931925.

and the construction of FFT circuits with reduced latency as mentioned in [5]. Ladner and Fischer [9] gave a general construction for parallel prefix circuits when gates have unbounded fan-out and Fich [5] provided new lower and upper bounds for circuits with gates having bounded fan-out. Snir [17] and Lakshmivarahan, Yang, and Dhall [10] derived upper and lower bounds on tradeoffs between circuit size and depth for parallel prefix circuits. In all of these papers [5], [9], [10], [17], the measure of complexity was the number of gates in the circuitry, rather than its VLSI area.

As in [5], we model the circuit as a directed acyclic graph, and assume without loss of generality that each node has fan-in and fan-out equal to 2. Each vertex represents one of the two types of gates—an operation or a duplication gate. An operation gate takes two inputs and produces the product of its two inputs while a duplication gate takes one input and produces multiple copies of its input. Thus, in the directed acyclic graph, every input is a vertex of in-degree 0, every operation gate is a vertex of in-degree 2 and fan-out at most two, and every duplication gate is a vertex of in-degree 1 and fan-out at most two. We define the depth of a vertex as the length of the longest path from any output to that vertex while the depth of the parallel prefix circuit is the maximum depth of any vertex in that circuit. An example of a parallel prefix circuit whose gates have fan-out at most two is shown in Fig. 1.

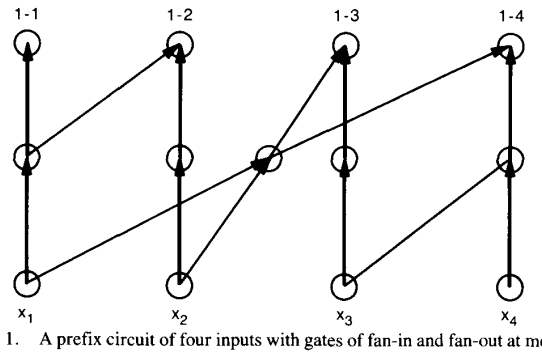


Fig. 1. A prefix circuit of four inputs with gates of fan-in and fan-out at most two.

III. Carry Look-Ahead Addition

It is well known that the delay time of a standard ripple-carry adder can be dramatically decreased by employing the scheme of carry look-ahead addition. The resulting adder circuitry has constant delay time, but contains certain gates whose fan-in is unbounded and certain inputs whose fan-out is unbounded. Carry look-ahead adders result from expanding the recurrence equation that describes the set of carries generated by the adder circuitry. If we start with

$$c_i = (a_i + b_i)c_{i-1} + a_i b_i$$
$$s_i = a_i \oplus b_i \oplus c_{i-1}$$

as the equations describing the ith carry and sum bits, and define generate and propagate variables $g_i = a_i b_i$, $p_i = a_i + b_i$, then the simple first-order linear recurrence $c_i = p_i c_{i-1} + g_i$ describes the problem of computing the carries. A carry look-ahead adder results from expanding out the above equation for each carry to obtain

$$c_i = g_i + \sum_{j=0}^{i-1} \left(\prod_{k=j+1}^{i} p_k \right) g_j.$$

Furthermore, the simple linear recurrence describing the computation of the carries can easily be shown to be equivalent to prefix computation. Inputs to the prefix computation are ordered pairs (p_i, g_i) and the associative operator $\circ$ is defined as $(p_i, g_i) \circ (p_j, g_j) = (p_i p_j, p_j g_i + g_j)$. With these definitions, the ith prefix of the ordered pairs is easily seen to contain as its second term the ith carry. There are many problems that arise when it is attempted to implement a carry look-ahead adder in VLSI using the above expansion. First of all, there are many multiinput gates contained in the resulting circuitry. Some VLSI technologies may support this with constant delay time. For example, NMOS does, as long as the logic equations are expressed in NOR–NOR form and the number of inputs is not too large so as to cause delays due to wiring capacitance and resistance. However, in other technologies, delay time may be proportional to the number of inputs to the gate, for example CMOS. To alleviate this problem, each multiinput gate can be replaced with a balanced tree structure that has bounded fan-in to each node. The modified circuitry then has logarithmic delay time.

Another problem with the resulting circuitry for a carry look-ahead adder is the area required to lay it out. Each carry requires a total of $1 + 2 + 3 + \cdots + i = i(i+1)/2$ inputs to its gates, so that area $\Omega(i^2)$ is required to realize it (note that this ignores interconnection complexity!). The area required to realize all the carries is thus $\Omega(i^3)$. A fundamental reason for this large amount of area consumption is that each carry directly generates all the subcomputations that it requires, so that a lot of duplicate work is performed among all the carries. If we are to reduce the area consumption of the circuitry, then we must somehow avoid doing all this duplicate work.

In this paper, by considering the more general problem of prefix computation we are able to derive upper and lower bounds on the area of VLSI circuit implementations for values of delay time between $\log \eta$ and $2 \log \eta$. Since prefix computation and binary addition are equivalent problems (a circuit for one can be made into a circuit for the other simply by changing the operation performed at each node), the bounds apply directly to carry look-ahead adders implemented in VLSI. We derive our bounds under the assumption of bounded fan-in and fan-out for each node in the circuitry. This has been a standard assumption in previous work in VLSI. In a practical sense, it is justified for certain technologies where a multiinput gate has delay proportional to the number of inputs, and a multioutput gate must by default drive a large capacitive load, resulting in delay time at best proportional to the logarithm of the number of outputs. Thus, the results presented here do have implications in a practical sense, and serve to give the designer of a carry look-ahead adder a sense of the optimality of his/her circuitry.

IV. The Model

Several models for VLSI circuits have been suggested by different authors during their investigations in establishing

lower bounds for a number of different problems in VLSI [1], [2], [4], [11], [12], [16], [19], [21], [22]. Our model is similar to the one proposed by [19] and used by [11]. The VLSI chip is abstracted as a rectangular grid which is a collection of horizontal and vertical tracks spaced apart at unit intervals. We embed a graph G in a grid by assigning nodes of G to points in the grid where horizontal and vertical tracks intersect. The edges of G correspond to paths in the grid which are restricted to follow along grid tracks but are not allowed to overlap for any distance. The paths may not cross any nodes to which they are not incident. The layout area of the embedding is then the product of the number of vertical tracks and the number of horizontal tracks which contain a node or path segment of the graph [11].

Our motivation in restricting our attention to circuits having gates of bounded fan-out arose from the constraints imposed by VLSI technology. Unbounded fan-out is possible in VLSI, but only at the expense of delay logarithmic in the size of the fan-out [13]. Thus, any implementation in VLSI of prefix computation requires delay time $\Omega(\log \eta)$ since the first input must fan-out to all outputs. Hence, circuits which make use of unbounded fan-out gates can be looked upon as circuits with gates of bounded fan-out with the same delay-time performance.

V. Lower Bounds on Area

Lower bounds on area as a function of computation time can be obtained in several ways for VLSI circuits [20]. One typical way is to consider the information flow from a subset of input parameters to the outputs. Another method to establish a lower bound on the area is to calculate the *minimum bisection width* (MBW) of the graph. The MBW of the graph is defined as the minimum number of edges ω which have to be removed in order to separate a set of nodes S into two sets S_1 and S_2 of almost equal size. Then using a result of [19], which states that area $= \Omega(\omega^2)$, the lower bound on area is immediately obtained. Unfortunately, the computation of MBW in general is difficult and the problem is known to be NP-complete [19]. In this paper, we use structural information associated with parallel prefix computation graphs to compute a lower bound on the MBW for all graphs which solve the parallel prefix problem within the assumed models. Briefly, our approach is as follows. We first extract a subgraph from the parallel prefix problem and determine some of its properties, which we then use in the solution of the larger problem.

Let a graph H be defined as follows. It has η inputs $x_1, \cdots, x_\eta$, η outputs $y_1, \cdots, y_\eta$, and there is exactly one path between every input and output. The vertices have in-degree and out-degree at most two. Moreover, all the vertices in the graph compute functions of the form $x_{p1} \circ x_{p2} \circ \cdots \circ x_{pm}$, $p_1 < p_2 < \cdots < p_m$ and $m \leq \eta$. For such graphs we can prove the following property.

Lemma 1: Assume a graph H has the properties described above and depth $\log_2 \eta + \kappa$, $0 \leq \kappa \leq \log_2 \eta$ (Fig. 2). Then it requires the removal of at least $\eta/2^\kappa$ edges to separate all the η inputs from all the η outputs.

Proof: Consider the case when $\kappa = 0$. Then every output y_i is the root of a complete binary tree with η leaves which are the η inputs x_j, $1 \leq j \leq \eta$. It follows then that every node of this complete binary tree has two inputs. Thus, at depth $= 1$ there will be two edges emanating from each output, implying the existence of at least $2\eta/2 = \eta$ nodes at that depth. By induction, every depth will have at least η nodes. Assume now that by the removal of $\beta < \eta$ edges we have been able to disconnect all input nodes from all output nodes. The proof proceeds by contradiction. Since removal of an edge can at most disconnect one node from the circuit, it follows that every level contains at least one node which is connected to two nodes each at the preceding and succeeding levels. It can now be seen that a path will always exist from an input to an output if only $\beta < \eta$ edges are removed (note that this is not the MBW of the graph H).

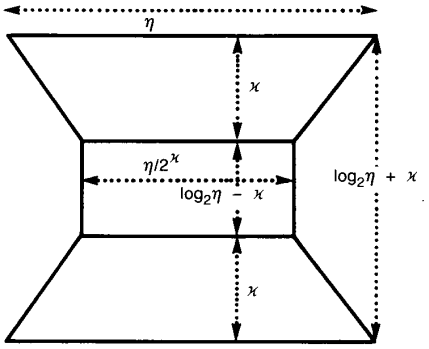


Fig. 2. The graph H of η inputs and depth $\log \eta + \kappa$.

For $1 \leq \kappa \leq \log_2 \eta$ the proof proceeds by induction. Consider the level at distance κ from the outputs. Because the reduction in the connections at every level to the next can at most reduce by two there will be at least $\eta/2^\kappa$ nodes at this level. Due to similar arguments, there will be at least $\eta/2^\kappa$ nodes at a distance κ from the inputs. Now these "information" nodes have to be connected to each other and have depth of at most $\log_2 \eta - \kappa$ to do so. This reduces to a case of $\eta' = \eta/2^\kappa$ nodes being in a graph H' of depth $\log_2(\eta/2^\kappa)$. Thus, at least $\eta/2^\kappa$ edges have to be removed in order to disconnect all the inputs from the outputs of the graph H. Q.E.D.

Calculation of the MBW of the Graph G

The strategy of this subsection is to consider several ways of assigning the nodes of the parallel prefix computation graph G to two sets S_1 and S_2 of almost equal size such that the number of edges E required to separate the nodes in the two sets is minimized. In order to calculate a lower bound on the MBW, we may select any set of nodes S, which we then divide into two sets S_1 and S_2 of equal size. For our purposes, it suffices to consider the set S as consisting of all the input and output nodes of the graph G. We shall show that the assignment of nodes to S_1 and S_2 which minimizes E also causes $\theta(\eta)$ inputs and outputs of the graph H (described earlier) to become bisected. The lower bound on the MBW of the graph G is then obtained by invoking Lemma 1. As it turns out, the "optimal" assignment of nodes of G is intimately linked to the distribution of pairs (x_i, y_i) defined as follows.

Let the parallel prefix computation graph G of η inputs x_1,

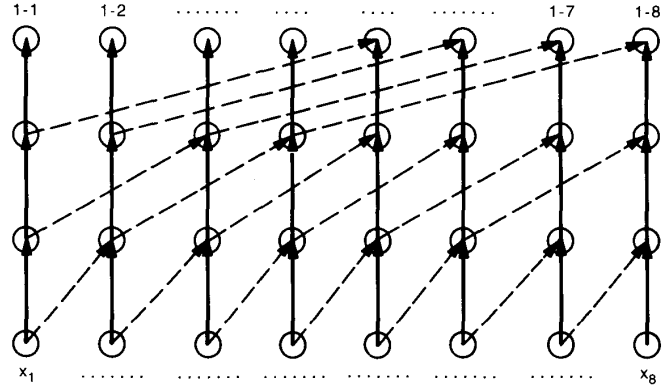


Fig. 3. A prefix circuit based on Kogge and Stone's solution for evaluating recurrence when the number of inputs is a power of two.

$\cdots, x_\eta$ and η outputs $y_1, \cdots, y_\eta$ have depth $\log_2 \eta + c$, $0 \leq c \leq \log_2 \eta$. Define a *pair* as $(x_i\ y_i)$, $i = 1, \cdots, \eta$. We *bisect* a pair (x_i, y_i) if x_i is assigned to S_1 and y_i is assigned to S_2 (or vice versa).

Before we proceed to the main theorem it is convenient to give two propositions which we will use in the proof.

Proposition 1: It takes the removal of β edges to bisect β pairs.

Proof: If $\alpha < \beta$ edges are required to bisect β pairs, then there must be at least one edge by removal of which we bisect ≥ 2 pairs. Consider now the two pairs which are bisected. The two outputs corresponding to the two pairs must be functions of both the inputs (of the two pairs). But the output with the lower index i cannot be a function of an input of index $> i$ and the proof follows by contradiction. Q.E.D.

Proposition 2: Let there be a set S of η indexes numbered from 1 to η. Consider two sets S_1 and S_2 each containing $\eta/2$ indexes such that $S_1 \cup S_2 = S$. Then we can always find subsets S_1' in S_1 of $\eta/4$ elements and S_2' in S_2 of $\eta/4$ elements such that all elements of S_1' are either all greater or all smaller than all elements of S_2'.

Proof: Select the lowest $\eta/4$ elements of S_1. Let there be $\eta/4 - \gamma$ elements in S_2 all of which are greater than all elements selected in S_1. If $\gamma = 0$, then we are done. Assume $\gamma > 0$. It follows then that the remaining $\eta/4 + \gamma$ elements of S_2 are smaller than at least one of the $\eta/4$ elements selected in S_1. Since these elements were the smallest $\eta/4$ elements of S_1 we conclude that all the $\eta/4 + \gamma$ elements of S_2 are smaller than all the largest $\eta/4$ elements of S_1 and the lemma is proven. Q.E.D.

Theorem: The MBW of a parallel prefix circuit of $\eta = 2^m$ inputs for some $m > 1$ and depth $\log_2 \eta + c$ satisfies MBW $\geq \eta/2^{c+2}$.

Proof: Out of the η pairs (x_i, y_i) $i = 1, \cdots, \eta$, let the assignment of nodes to the sets S_1 and S_2 be such that α pairs are bisected. Then the remaining $(\eta - \alpha)/2$ pairs are distributed in each of the two sets. By Proposition 2, at least $(\eta - \alpha)/4$ inputs in S_1 have indexes lower than at least $(\eta - \alpha)/4$ inputs in S_2. Thus, at least $(\eta - \alpha)/4$ outputs in S_2 are functions of at least $(\eta - \alpha)/4$ inputs in S_1 implying the existence of a path from every one of these $(\eta - \alpha)/4$ inputs of S_1 to every one of the $(\eta - \alpha)/4$ outputs of S_2. These inputs and outputs form a circuit "H" discussed earlier, the value of "κ" now being $\log_2 4\eta/(\eta - \alpha) + c$. Using Lemma 1 and Propositions 1 and 2, the number of edges required to disconnect the nodes of the two sets S_1 and S_2 is $\geq \alpha + (\eta - \alpha)^2/\eta 2^{4+c}$. It follows then that the minimum is attained when α is 0 and the MBW is $\geq \eta/2^{4+c}$. Q.E.D.

Corollary: The area of a parallel prefix graph of $\eta = 2^m$ inputs for some $m \geq 1$ and time of computation $T \leq 2 \log_2 \eta$ satisfies AREA $= \Omega(\eta^2 2^{2(\log_2 \eta - T)} + \eta)$.

Proof: The first term in the expression comes from the observation that area $= \Omega(\omega^2)$, ω in this case being $\geq \eta/2^{c+4}$ and $c = T + \log \eta$. The second term is a lower bound on the number of nodes a prefix circuit must contain—which is simply η. Q.E.D.

Thus, we have the result that for time of computation T, $\log \eta \leq T \leq 2 \log \eta$, the VLSI area required for a prefix circuit of η inputs and η outputs is $\Omega(\eta^2 2^{2(\log_2 \eta - T)})$.

VI. Upper Bounds

In the light of the lower bound result of the previous section, it is of interest to discover circuits which are optimal in area for time of computation T, $\log \eta \leq T \leq 2 \log \eta$. Such constructions can then be directly used to generate uniform circuits for carry look-ahead adders in a way similar to the one employed by Brent and Kung [2]. Constructions similar to the ones presented here have been obtained independently by Han [6].

At the extremes of $T = \log \eta$ and $T = 2 \log \eta$, circuits are already known that achieve the lower bounds of Section V. For $T = \log \eta$, constructions of Kogge and Stone [8] can easily be interpreted to form prefix circuits that can be laid out in area $O(\eta^2)$. An example of such a circuit is given in Fig. 3. For $T = 2 \log \eta$, constructions of Brent and Kung [2] for regular adders (Fig. 4) achieve optimality if we impose the additional constraint that the inputs and the outputs lie on the boundary. Since their circuits can be laid out in area $O(\eta \log \eta)$, our results reveal an extreme area-time tradeoff behavior in the layout of prefix computation graphs. A constant factor reduction in time of computation for $2 \log \eta$ to $\log \eta$ results in an increase in area from $O(\eta \log\ )$ to $\Omega(\eta^2)$, and we can also

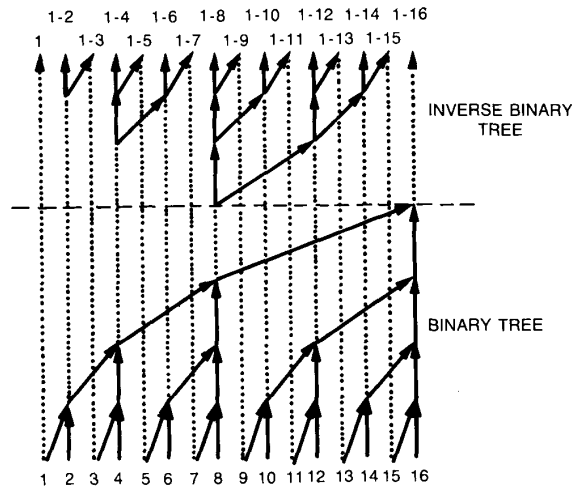


Fig. 4. Brent and Kung's construction.

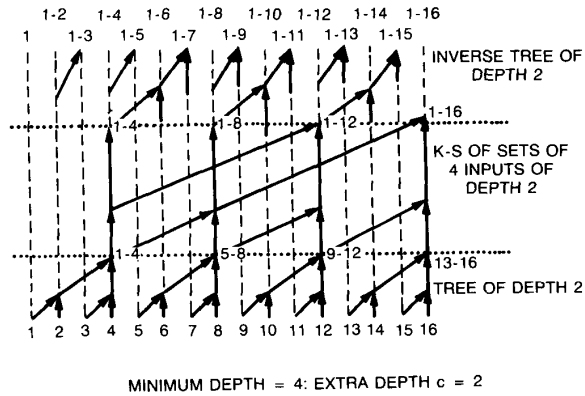


Fig. 5. A prefix circuit of 16 inputs and depth 6.

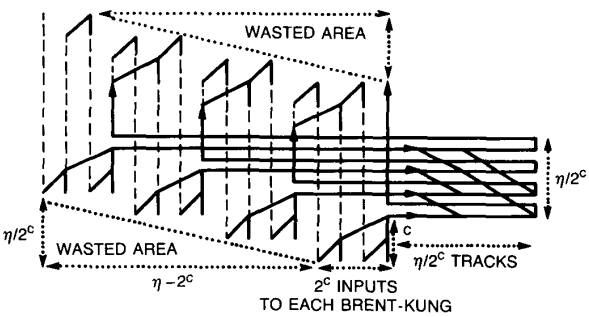


Fig. 6. A regular layout for prefix circuits with depth $\log \eta + c$, $c \leq \log \eta$.

conclude that the Kogge–Stone circuits are optimal in terms of their area.

Constructions for prefix circuits which are area-efficient for values of T between $\log \eta$ and $2 \log \eta$ may be obtained by combining the constructions of Brent–Kung and Kogge–Stone. Fig. 5 illustrates the way the two constructions are combined to compute prefixes. The Kogge–Stone circuitry on $\eta/2^c$ inputs/outputs takes the place of the H graph from Section V, and is surrounded by the two halves of the Brent–Kung circuitry. Such a combination is easily verified to compute prefixes correctly. The circuit can be made into a slightly more area-efficient layout by isolating the Kogge–Stone portion so that it consumes as little area as possible (see Fig. 6). When this is done, the area of the entire structure can by analyzed as follows. Let $c = T - \log \eta$ as in Section V. The number of horizontal units used in the large (left) portion of the graph is η. The number of vertical units used is $O(2(c + \eta/2^c))$. Both these expressions arise from considering the area requirement of the K–S circuitry. The wasted area indicated in Fig. 6 can be expressed as $O(\eta^2/2^c)$. It is justifiable to subtract this wasted area from the product of horizontal and vertical units since in practice it is available to the designer for the implementation of other VLSI structures. Doing this, we obtain an expression for the active area required by our layout: $O(c\eta + (\eta^2/2^c))$, as compared to the lower bound of $\Omega((\eta/2^c)^2)$ derived in Section V.

One interesting facet of our upper and lower bounds on area is that for time of computation T, $2 \log \eta - \text{loglog } \eta \leq T \leq 2 \log \eta - 1$, the area requirements reduce to $O(\eta \log \eta)$. Thus, if we employ matching constructions, it would imply that we could have an improvement in delay time of loglog η over Brent and Kung's [2] constructions without any asymptotic penalty in area. In practice, this observation could lead to the implementation of area-efficient adder structures in VLSI that are faster than those previously known (e.g., [2]).

If we permit the input and output nodes to be placed inside the circuit (instead of insisting they be on the boundary), then there is an optimal layout which matches the derived lower bound to within a constant factor. This layout, shown in Fig. 7, again reduces the area consumption of the K–S circuitry to be near minimal. The B–K circuitry is further broken up into subcircuits of 2^c inputs/outputs, each of which provides an input to and receives an output from the K–S circuitry. These multiple subcircuits are then stacked on top of each other, which minimizes the overall number of horizontal units used. One wire connects the combination of the inputs of a subcircuit to the K–S circuitry, and an output of the K–S circuitry is fed back to the middle of the subcircuit where it is then "fanned out" to all outputs. Additional horizontal units for these wires must be provided, which causes the horizontal dimension to grow by a constant factor of ≤ 3. The number of horizontal units in this layout is $O(\eta/2^c)$. Since there are $O(\eta/2^c)$ B–K subcircuits each of height $2c - 1$, the number of vertical units is $\leq O(c\eta/2^c)$ (more than one B–K subcircuit may be placed at the same level). Thus, the area taken is $O(c\eta^2/2^{2c})$. This circuit achieves area $O(\eta \log \eta)$ for values of T in the range $3/2 \log \eta \leq T \leq 2 \log \eta - 1$. Hence, asymptotically we have an additional decrease in delay over Brent and Kung's constructions for area consumption $O(\eta \log \eta)$.

VII. Discussion

In this section, we shall briefly discuss the effect of varying the underlying models on the applicability of our results. First, we may mention at the outset that increasing the fan-in and fan-out of the vertices and nodes (of the prefix circuit and VLSI grid model) from 2 to f corresponds to taking logarithms to the base f in our lower bound. Hence, for the case of

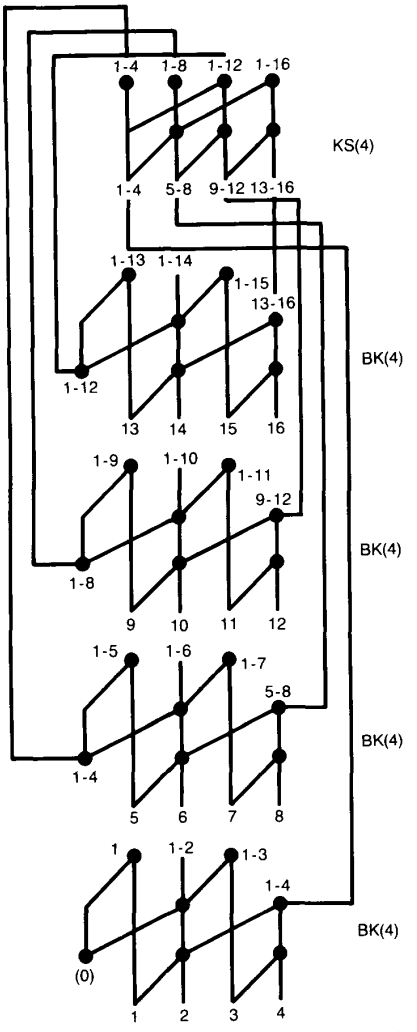


Fig. 7. Area optimal layout ($k = 2$, $n = 16$).

bounded fan-in and fan-out, our lower bound would be $\Omega(\eta^2 f^{2(\log_f \eta - T)} + \eta)$, $\log_f \eta \leq T \leq 2 \log_f \eta$. However, if we permit the vertices and nodes in our models of a prefix circuit and a VLSI grid to have unbounded fan-out, the MBW of the graph falls to $\Omega(1)$ and the lower bound becomes $\Omega(\eta)$.

Also, we have not accounted for delay penalties due to wire capacitances, and instead have assumed that all wires in our circuitry have unit delay. In practice, this is not true for VLSI circuits corresponding to extremely large values of η, and thus the tradeoffs shown here would not be observed in appropriate models [4] for very large η.

To conclude this section, we mention that the interesting aspect of the extreme area–time tradeoff behavior is that it occurs at all.

VIII. Conclusion

In this paper, we discussed the VLSI layout of prefix computation graphs. For computation time T, $\log \eta \leq T \leq 2 \log \eta$, we demonstrated an "unusual" behavior. A constant factor decrease in computation time implied a nonconstant increase in area from $O(\eta \log \eta)$ to $\Omega(\eta^2)$. Also, for every unit decrease in delay time below $2 \log \eta$, area was observed to double in value.

Since the carry computation of a binary adder can be reduced to prefix computation [3], we concluded that this behavior would be exhibited by a carry look-ahead adder implemented in VLSI.

It seems that similar behavior could exist for other problems than the one considered here. Thus, it would be interesting to pursue this line of research more thoroughly, as such lower bounds would be of practical significance to a VLSI designer trying to discover efficient layouts for circuits of minimum/near-minimum depth.

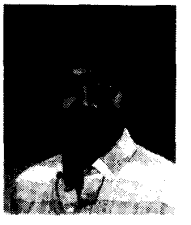
Acknowledgment

The authors would like to thank W. Evans and the referees for their careful review of the paper.

References

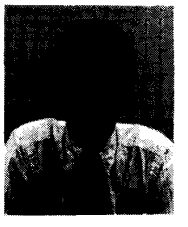
[1] R. P. Brent and L. M. Goldschlager, "Some area-time tradeoffs for VLSI," *SIAM J. Comput.*, vol. 11, no. 4, pp. 737-747, Nov. 1982.

[2] R. P. Brent and H. T. Kung, "A regular layout for parallel adder," *IEEE Trans. Comput.*, vol. C-31, no. 3, pp. 260-264, Mar. 1983.
[3] A. Chandra, S. Fortune, and R. Lipton, "Lower bounds for constant depth monotone circuits for prefix functions," in *Proc. 10th Int. Conf. Automata, Languages, Programming,* 1983, pp. 109-117.
[4] B. Chazelle and L. Monier, "A model of computation for VLSI with related complexity results," in *Proc. 13th ACM Symp. Theory Comput.*, 1981, pp. 318-325.
[5] F. E. Fich, "New bounds for parallel prefix circuits," in *Proc. 15th ACM Symp. Theory Comput.*, 1983, pp. 100-109.
[6] T. Han, personal communication, 1986.
[7] R. B. Johnson, "The complexity of a VLSI adder," *Inform. Processing Lett.*, vol. 11, no. 2, pp. 92-93, Oct. 1980.
[8] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Trans. Comput.*, vol. C-22, no. 8, pp. 786-793, Aug. 1973.
[9] R. E. Ladner and M. J. Fischer, "Parallel prefix computation," *J. ACM,* vol. 27, no. 4, pp. 831-838, Oct. 1980.
[10] S. Lakshmivarahan, C. Yang, and S. K. Dhall, "On a new class of optimal parallel prefix circuits," in *Proc. 1987 Int. Conf. Parallel Processing,* Aug. 1987, pp. 58-65.
[11] F. T. Leighton, "Layouts for the shuffle-exchange graph and lower bound techniques for VLSI," Ph.D. dissertation, Mathematics Dep., Massachusetts Instit. Technol., Aug. 1981.
[12] C. E. Leiserson, "Area efficient VLSI computation," Ph.D. dissertation, Dep. Comput. Sci., Carnegie-Mellon Univ., Nov. 1980.
[13] C. A. Mead and L. A. Conway, *Introduction to VLSI Systems.* Reading, MA: Addison-Wesley, 1980.
[14] K. Mehlhorn and F. P. Preparata, "Area-time optimal VLSI integer multiplier with minimum computation time," *Inform. Contr.*, vol. 58, pp. 137-156, 1983.
[15] Y. Ofman, "On the algorithmic complexity of discrete functions," *Cybern. Contr. Theory, Soviet Phys. Doklady,* vol. 7, pp. 589-591, Jan. 1963.
[16] J. E. Savage, "Planar circuit complexity and the performance of VLSI algorithms," in *VLSI Systems and Computations,* H. T. Kung, B. Sproull, and G. Steele, Eds. Rockville, MD: Computer Science Press, 1981, pp. 61-68.
[17] M. Snir, "Depth-size tradeoffs for parallel prefix computation," *J. Algorithms,* vol. 7, no. 2, pp. 185-201, 1986.
[18] B. Sugla, "Parallel computation using limited resources," Ph.D. dissertation, Dep. Elec. Comput. Eng., Univ. of Massachusetts, Amherst, MA, July 1985.
[19] C. D. Thompson, "A complexity theory for VLSI," Ph.D. dissertation, Carnegie Mellon University, Dep. Comput. Sci., 1980.
[20] J. Ullman, *Computational Aspects of VLSI.* Rockville, MD: Computer Science Press, 1983.
[21] J. Vuillemin, "A combinatorial limit to the computing power of VLSI circuits," in *Proc. 21st IEEE Symp. Foundations Comput. Sci.,* Nov. 1980, pp. 294-300.
[22] A. C. Yao, "The entropic limitations on VLSI computations," in *Proc. 13th ACM Symp. Theory Comput.*, 1981, pp. 308-311.



Binay Sugla (S'83-M'85) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, in 1981 and the Ph.D. degree in electrical and computer engineering from the University of Massachusetts, Amherst, in 1985.

Currently, he is with AT&T Bell Laboratories, Holmdel, NJ. He has published in and is interested in the areas of parallel algorithms, parallel processing systems, distributed routing, VLSI circuit design, and optical computing.



David A. Carlson (S'80-M'80) received the B.A. degree in mathematics (summa cum laude) from the University of Connecticut, Storrs, in 1977, and the Ph.D. degree in computer science from Brown University, Providence, RI, in 1980.

He is currently a Research Staff Member at the Supercomputing Research Center, Bowie, MD. From 1980 to 1986, he was an Assistant Professor of Electrical and Computer Engineering at the University of Massachusetts, Amherst. His research interests include the design and analysis of algorithms, parallel processing, VLSI, and concrete computational complexity, and he has published over 30 technical papers on these subjects.

Dr. Carlson is a member of the IEEE Computer Society, the Association for Computing Machinery, and the Society for Industrial and Applied Mathematics.