



**Centrum voor Wiskunde en Informatica**  
Centre for Mathematics and Computer Science

---

G. Kissin

Upper and lower bounds on switching energy in VLSI

Computer Science/Department of Algorithmics & Architecture

Report CS-R9044

September

---

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Research (N.W.O.).

# UPPER and LOWER BOUNDS on SWITCHING ENERGY in VLSI

*Gloria Kissin*

Centre for Mathematics and Computer Science  
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

and

Department of Mathematics and Computer Science  
University of Amsterdam, Plantage Muidersgracht 24  
1018 TV Amsterdam, The Netherlands

September 1990

## ABSTRACT

A technology independent framework is established for measuring the *switching energy* consumed by very large scale integrated (VLSI) circuits. Techniques are developed for analyzing functional energy consumption, and for designing energy-efficient VLSI circuits. A wire (or gate) in a circuit uses switching energy when it changes state from 1 to 0 or vice versa. This paper develops the *Uniswitch Model (USM)* of energy consumption, which measures the differences between pairs of states of an embedded circuit.

The following worst case lower bounds are obtained in *USM*. *Monotone* circuits require switching energy proportional to the circuit's area. A class of  $n$ -input, boolean valued functions, including addition and multiplication, uses  $\Omega(n \log_2 n)$  switching energy, when computed by a shallow depth circuit. A special case of the parity function is shown to require switching energy proportional to the area.

This paper also derives upper bounds in *USM*. Novel circuits and layouts are obtained for  $n$ -bit *OR* and compare functions that have shallow depth and use only linear energy, in the worst case. A shallow depth  $n$ -bit addition circuit is laid out in a novel manner that uses linear energy, on the average. This is a log factor better than the worst case lower bound for addition.

**1980 Mathematics Subject Classification:** 68A30

**CR Categories:** B.7.1, F.1.1

**Key Words and Phrases:** addition, compare functions, circuits, OR, parity, switching energy, uniswitch energy, uniswitch model, upper and lower bounds, USM, VLSI.

**Note:** This paper will appear in a forthcoming issue of *JACM*. This paper was partially prepared at *CSRI, University of Toronto*.

Report CS-R9044  
Centre for Mathematics and Computer Science  
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands



## 1. INTRODUCTION

This paper establishes upper and lower bounds on the switching energy required to realize some commonly computed functions with VLSI circuits. Specifically, OR and AND functions and compare functions are shown to be realizable with VLSI circuits that use an amount of switching energy that is never more than linear in the length of the input. A linear *average* energy layout is given for binary addition. Worst case lower bounds are obtained for monotonic circuits, a class of multiple-output functions including addition and multiplication, and for a special case of the parity function.

Energy is practically motivated in VLSI design because energy consumed by a circuit is transformed into heat. How well a circuit can dissipate heat determines its operational limitations. Thus, the less heat produced the better. In addition, energy considerations determine a significant portion of the overall costs of a computer [Me86].

Common to all physical devices is the switching energy [MC80] consumed when a wire or gate changes state from 1 to 0 or vice versa. The amount of switching energy consumed is proportional to the area switched.

The *Uniswitch Model (USM)* of energy consumption, defined in the next section, and most results in this paper first appeared in [Ki82] and [Ki85]. This work also comprises part of [Ki87]. Lengauer and Mehlhorn [LM81] showed that  $n$ -input functions realizable in  $AT^2 = O(n^2)$  require  $\Omega(AT)$  switching energy, where  $A$  is *area* and  $T$  is *time* in the Thompson model [Th80]. Aggarwal et al [ACR88] improved the result of Lengauer and Mehlhorn to obtain an  $\Omega(n^2)$  energy bound for the class of *transitive* functions [Vu83]. Leo [Le84] independently showed that, for a specialized circuit basis, the parity function requires  $\Omega(A)$  average switching energy, where  $A$  is the area of the parity circuit.

The bounds obtained in this paper estimate the wire switching energy in *USM*. Neglecting node switching energy is not a major restriction because nodes are assumed to have minimal area and circuits are generally connected graphs. *USM* measures the differences between two states of a circuit; hence at most one switching event per node or wire is recorded by *USM*. Race conditions (aka hazards) that induce wires to switch more than once are the domain of the *Multiswitch Models*, which are defined and discussed in [Ki87] and [Ki90]. *USM* provides a lower bound on the total energy used by a circuit.

The rest of this paper is organized as follows. Section 2 defines the *Uniswitch Model* of energy consumption, aka *uniswitch energy*. (The term *energy* refers to wire switching energy for the duration of this paper). Two worst case energy metrics are defined and shown to be equivalent up to a constant multiplicative factor. The motivation for *USM* is discussed in section 2.1.

Section 3 examines energy lower bounds in *USM*. Monotone circuits (ie. basis  $\{\wedge, \vee\}$ ) are shown to be inherently *energy-inefficient* in that a monotone circuit always uses worst case uniswitch energy proportional to the circuit's area. Nonlinear lower bounds are obtained for a class of multiple output functions including addition and multiplication. For a restricted basis, a nonlinear lower bound is obtained for the single output function, parity. All the bounds are in the form of *energy-time* tradeoffs, because allowing large time bounds (eg. linear circuit depth) often permits trivial energy upper bounds.

In section 4, upper bounds are obtained in *USM*. In particular, a novel circuit and layout is described for the *OR* function (ie.  $x_1 \vee x_2 \vee \dots \vee x_n$ ), which is optimal in expending  $O(n)$  worst case uniswitch energy. The *OR* technique is extended to compare functions (ie. Is  $X > Y$ ? where  $X, Y \in \{0,1\}^n$ ).

Section 5 addresses the question of average case energy consumption. A parallel prefix circuit for binary addition, laid out according to the technique of section 4, is shown to use linear uniswitch energy

on the average. This is a log factor better than the worst case lower bound. Some open problems are stated in section 6.

### 2. THE SETTING

The *Uniswitch Model* of energy consumption defines an energy cost measure for VLSI circuits. *USM* measures the differences between pairs of states of a circuit. The following discussion sets the stage for a precise definition of *USM*.

A *VLSI circuit* is a combinational circuit [Bo77] embedded in a plane as in [BK81]. Salient assumptions of the Borodin / Brent / Kung models that are important to *USM* are as follows. A circuit is acyclic. A wire (edge) in a VLSI circuit has constant minimum width  $\lambda > 0$ . A non-input node (gate) in a circuit computes a logical function of 1 or 2 inputs (eg. AND ( $\wedge$ ), OR ( $\vee$ ), NOT ( $\neg$ )) in constant time. Gates are separated by distances  $\geq \lambda$ . A gate has constant area  $\lambda^2$ , and a non-output gate has fanout 1 or 2. Input nodes have fanin 0. Output nodes have fanout 0. At most a constant number of wires,  $v \geq 2$ , can overlap or intersect at any point in a VLSI circuit.

Some of the energy lower bounds described in this paper are functions of the wire area of a VLSI circuit. These results require that the circuit in question be connected, that the circuit inputs preclude constant values, and that interior nodes are functionally dependent on the inputs (ie. each interior node has two states).

#### Definition:

A *CID VLSI circuit* is a VLSI circuit that satisfies the following two properties.

**E1:** Each input has fanout at most  $p$ , for  $p \geq 1$ . This limits the number of *free* duplicates of any input bit. Presumably, a real circuit receives one or a few instances of an input bit, and if more are needed the input must be replicated by the circuit. This costs area and energy that cannot realistically be neglected. In section 2.1, an example is given that illustrates the asymptotic affect of replicating inputs.

**E2:** All instances of any input appear at input nodes that are within a constant distance of each other. This again is an input fanout constraint. For the purposes of this paper, input and output nodes are on a convex boundary of the circuit layout.

Properties E1 and E2 are called *constant input duplicates (CID) assumptions*. This paper is primarily concerned with CID VLSI circuits and their physical analogs. Hence, the term *circuit* generally refers to a CID VLSI circuit. The terms *real circuit* or *physical circuit* refer to the physical analog of a CID VLSI circuit. (In [Ki87], CID circuits are called CIF circuits.)

#### Definitions:

A *legal state*, (hereafter, also called *state* or *stable state*)  $s$ , is a function that attributes values to the nodes and wires of a circuit  $C$ . ie.  $C = (V, W)$  where  $V$  is the node set and  $W$  is the set of wires.  $s: V \cup W \rightarrow \{0,1\}$ . Input node  $x$  has some value  $x_0$  where  $x_0 \in \{0,1\}$ . Edge  $w$  emanating from input node  $x$  has value  $s(w) = x_0$ . Non-input nodes and edges have values consistent with the input and the labeling of the nodes (eg.  $s(\text{AND}(0,1)) = 0$ ,  $s(\text{NOT}(0)) = 1$ ).  $s_X$  denotes the state of  $C$  for input  $X$ .  $X \rightarrow s_X$  is a bijection between an input vector and a state of circuit  $C$ . Since a state and its associated input vector are closely allied, they are used interchangeably in the following discussion.  $C$  is in state  $s_i$  at time  $t_i$ .  $s_0$  is the *initial state* of  $C$ .

The switching energy of a circuit  $C$  is defined on a pair of states. In particular, we are interested in what happens when one input vector to  $C$  is replaced by another input vector. In the definitions that follow, the pair of states in question is often denoted as  $(s_0, X)$ , where  $s_0$  is the initial state and  $X$  is an

## Upper and Lower Bounds on Switching Energy in VLSI

input vector that induces a second (ie. final) state.

### Definitions:

Suppose VLSI circuit  $C$  changes state from  $s_0$  to  $s_f$ , denoted  $C : s_0 \rightarrow s_f$ . Further assume that wire  $w$  has initial value  $s_0(w) = w_0$  and final value  $s_f(w) = w_f$  where  $w_0, w_f \in \{0,1\}$ . This change in the value of  $w$  is denoted  $w : w_0 \rightarrow w_f$ . Then  $w$  is *switched* (*switches*) iff  $w_0 \neq w_f$ . A wire of length  $L$  that switches accounts for  $L/k$  *switching energy*, where  $k > 0$  is the length of wire that accounts for 1 unit of switching energy. If  $W = \{w\}$  is the set of wires in circuit  $C$ , and  $X$  is the input set such that  $C : s_0 \rightarrow X$ , then the *wire energy*,  $E_w$ , consumed by  $C$  is  $E_w(C, s_0, X) \triangleq \frac{1}{k} \left( \sum_{\substack{w \in W \\ s_0(w) \neq s_X(w)}} \|w\| \right)$ ,

where  $\|w\|$  is the area of wire  $w$ .  $E_w(C, s_0, X) \leq \frac{1}{k} * \text{area}(C)$  where  $\text{area}(C)$  is the total wire area of  $C$ .

*Uniswitch energy* is defined below for the worst case and average case. Two worst case uniswitch energy models, denoted  $E_{worst}^U$  and  $E_{worst}^L$ , are defined below.  $E_{worst}^U$  picks the maximum wire energy expended when all (initial state, input) pairs are considered. This is the most appropriate measure for analyzing circuits - hence the superscript  $U$  for *Upper* bounds.  $E_{worst}^L$  is obtained by first determining the maximum energy expenditure over all inputs for each initial state. From this set of maxima, the minimum is chosen to obtain a bound that is valid for all initial states.  $E_{worst}^L$  thus yields strong *Lower* bounds - hence the superscript  $L$  - independent of a circuit's initial state.

The average energy model  $E_a$ , defined below, averages the wire energy expended by a circuit over all (initial state, input) pairs.

### Definitions:

If  $C_n$  is a VLSI circuit computing  $f_n : \{0,1\}^n \rightarrow \{0,1\}^m$  such that  $C_n$  is in state  $s_0$  at time  $t_0$ , and  $E_w(C_n, s_0, X)$  is the wire energy consumed by  $f_n$  when  $X = (x_1, \dots, x_n)$  is the input to  $C_n$  at time  $t > t_0$ , then  $E_{worst}^L(C_n)$ , the *universal worst case uniswitch energy*, is given by

$$E_{worst}^L(C_n) \triangleq \min_{s_0} [\max_X E_w(C_n, s_0, X)]$$

$E_{worst}^U(C_n)$ , the *existential worst case uniswitch energy*, is given by

$$E_{worst}^U(C_n) \triangleq \max_{(s_0, X)} E_w(C_n, s_0, X)$$

and  $E_a(C_n)$ , the *average case uniswitch energy* is given by

$$E_a(C_n) \triangleq \sum_{(s_0, X)} E_w(C_n, s_0, X) / 2^{2n}$$

where  $2^{2n}$  is the number of  $(s_0, X)$  pairs. This definition of  $E_a(C_n)$  assumes that the input vector is uniformly distributed over  $\{0,1\}^n$ .

$E_{worst}^U(C_n)$  is abbreviated as  $E_{worst}^U$ .  $E_{worst}^L(C_n)$  is abbreviated as  $E_{worst}^L$ .

$E_{worst}^L(C_n)$  is a good model for lower bound analyses, while  $E_{worst}^U(C_n)$  seems better suited for upper bounds. The following theorem, however, shows that the two models are equivalent to within a constant factor. Therefore, it is sufficient to consider  $E_{worst}^U$  for both upper and lower bounds, and  $E_{worst}^U(C_n)$  will often be abbreviated as  $E_{worst}$ .

### Theorem 2.1:

$$E_{worst}^U \geq E_{worst}^L \geq \frac{1}{2} E_{worst}^U$$

**Proof:**

The first inequality,  $E_{worst}^U \geq E_{worst}^L$ , is obvious.

Consider a function  $f_n$  for which a circuit  $C_n$  uses  $E_{worst}^U = B$ . Thus, by definition of  $E_{worst}^U$ , there exists two states of  $C_n$ ,  $s_1$  and  $s_2$  such that when  $C_n$  switches from  $s_1$  to  $s_2$ ,  $B$  amount of energy is consumed. Let  $A$  be the subset of wires of  $C_n$  that have different values in  $s_1$  and  $s_2$ . ie.  $C_n = (V, E)$  where  $E$  is the set of wires, and  $A = \{w \mid w \in E \text{ and } s_1(w) \neq s_2(w)\}$ .

By definition,  $\text{area}(A) = B$ .

Assume  $C_n$  is in some state  $s$  other than  $s_1$  or  $s_2$ . Consider the set  $A$  of wires. Let  $A_1 = \{w \mid w \in A \text{ and } s(w) = s_1(w)\}$

Let  $A_2 = A - A_1 = \{w \mid w \in A \text{ and } s(w) = s_2(w)\}$

There are two possible cases:

case 1:  $\text{area}(A_1) \geq \frac{1}{2} \text{area}(A)$

Apply the appropriate inputs that will induce  $s_2$  (ie. cause  $C_n$  to switch from state  $s$  to state  $s_2$ ).

This will cause the wires of  $A_1$  to switch, thereby using energy  $\geq \frac{1}{2} B$ .

case 2:  $\text{area}(A_2) \geq \frac{1}{2} \text{area}(A)$

Apply the appropriate inputs to induce state  $s_1$ . This will cause the wires of  $A_2$  to switch, thereby using energy  $\geq \frac{1}{2} B$ .

Since the argument above applies to an arbitrary function  $f_n$ , an arbitrary circuit  $C_n$  for  $f_n$ , and an arbitrary state  $s$  of  $C_n$ , it follows that  $E_{worst}^L \geq \frac{1}{2} E_{worst}^U$ . []

**Definitions:**

A function  $f: \{0,1\}^* \rightarrow \{0,1\}^*$  is *energy efficient* iff  $\exists$  a family  $C = (C_n)_{(n \in \mathbf{N})}$  of circuits with  $C_n$  realizing  $f \upharpoonright \{0,1\}^n$ , and  $E_{worst}(C_n) = \Theta(n)$ . Circuit family  $C = (C_n)_{(n \in \mathbf{N})}$  is *energy efficient* iff  $\forall$  families  $\hat{C} = (\hat{C}_n)_{(n \in \mathbf{N})}$  of circuits with  $\Phi(\hat{C}) = \Phi(C) : E_{worst}(\hat{C}_n) = \Omega(E_{worst}(C_n))$ .

Throughout this paper,  $\log n$  means  $\log_2 n$ .

**2.1 Model Motivation**

The intent of this section is to motivate the *Uniswitch Model* in light of physical considerations, and to discuss and motivate some of the assumptions included in the definition of *USM*.

*USM* is a good model for obtaining lower bounds because it conservatively estimates a circuit's switching behaviour. Thus, a lower bound in *USM* is an equally valid lower bound on *multiswitch* energy.

*USM* takes no notice of how a circuit arrives at a particular state. This is the domain of the *Multiswitch Models*, which are discussed in [Ki87] and [Ki89]. However, in order to discuss the relevance of using *USM* to obtain upper bounds, the following multiswitch notions are introduced.

The switching behaviour of physical circuits is influenced by various delay functions, such as gate delay  $\delta$ , wire delay  $\Delta$  and input delay  $I$ .  $\delta$  determines the switching speed of a gate.  $\Delta$  determines the time to transmit a bit along a wire.  $I$  determines when an input value arrives at an input port.

**Definitions:**

Let  $(C_n, \delta, \Delta, I)$  denote a *circuit scheme*, where  $C_n$  is a CID VLSI circuit with gate delay  $\delta$ , wire



delay  $\Delta$ , and input delay  $I$ . A circuit scheme  $(C_n, \delta, \Delta, I)$  exhibits the *uniswitch property* if each node or wire of  $C_n$  switches at most once when  $C_n$  changes from one input setting to another, according to  $\delta, \Delta$  and  $I$ . Otherwise,  $(C_n, \delta, \Delta, I)$  exhibits the *multiswitch property*.

Using *USM* to obtain upper bounds is justified for circuit schemes that exhibit the uniswitch property. For example, if each node of a circuit receives its inputs at the same time, race conditions cannot arise. The uniswitch property is thus ensured. Some real circuits have this timing property. Where race conditions derive solely from a circuit's asynchrony (ie. the paths to a node vary in length), a circuit scheme can acquire the uniswitch property if the circuit can be made synchronous. A "bad" input schedule can be offset by varying gate delays. These approaches to designing circuit schemes that achieve the uniswitch property are discussed in [Ki87] and [Ki89]. Further, according to C. Mead [Me86], many CMOS designs are synchronized to ensure that the corresponding circuit schemes have the uniswitch property.

*USM* is the first step in the systematic asymptotic analysis of switching energy consumption in VLSI circuits. As such, *USM* is justified as an upper bound model. In addition, *USM* is motivated by designers' practical efforts to prevent hazards and thus ensure the uniswitch property.

The rest of this section discusses the following circuit assumptions.

- 1) Circuits are acyclic.
- 2) Each node of a VLSI circuit uses area  $\lambda^2$ .
- 3) Each input has at most constant fanout.
- 4) All instances of an input appear within constant distance of each other.

The first two constraints are simplifying assumptions, while the last two constraints attempt to model real circuits. The following elaborates on the four assumptions listed above.

- 1) Circuits are acyclic.

The study of combinational circuits (without loops) has a long and distinguished history. Krohn and Rhodes' [KR65ab] seminal work in this area showed that each sequential machine (with loops) can be decomposed into structures consisting of only combinational circuits and flip-flops.

A recommended architecture for sequential machines is the finite state machine in which the combinational logic is isolated from the looping structure [MC80]. See Figure 2.0. This architecture lends itself to analysis of the combinational logic distinct from the looping buffers.

- 2) Each node of a VLSI circuit uses area  $\lambda^2$ .

That the area of each node is at least  $\lambda^2$  is dictated by manufacturing constraints.  $\lambda$  is a technology dependent value that determines the minimum feature size of circuit components. Large node fanin requires node area proportional to the fanin. However, since our underlying circuit model has fanin at most two, minimal node area is sufficient. A constant fanin greater than two increases node area by a constant factor. Unbounded fanin is reasonably excluded from consideration since practical circuits consist of components with bounded fanin.

A node requires large area to drive a large capacitive load, such as a large fanout or a long wire, without a degradation in time. Our underlying circuit model assumes that a node has outdegree at most two. Realizing a constant fanout greater than two increases the depth and area of the circuit by a constant factor. A larger fanout can be realized with a binary tree. Wire delay is not a concern of *USM*, which measures the differences between two static states.

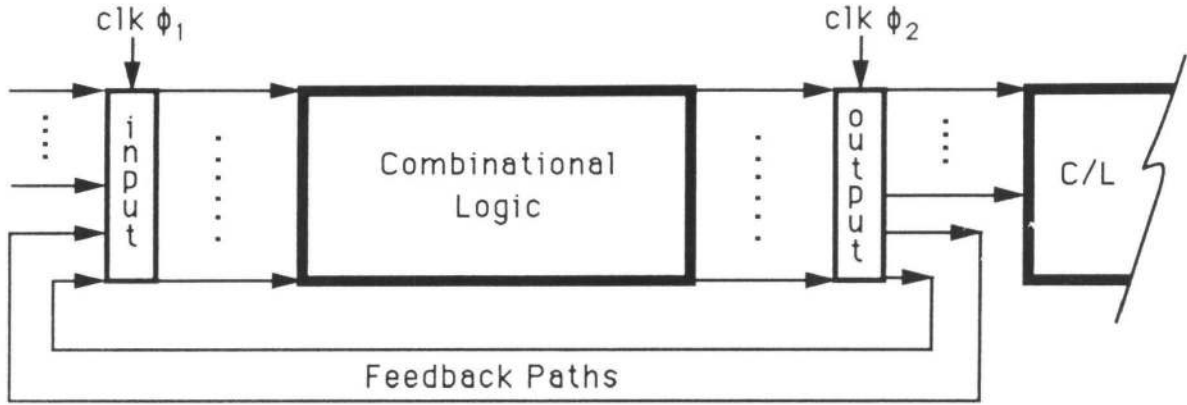


Figure 2.0. Preferred Sequential Circuit: a Finite State Machine

I/O ports will generally be much larger than  $\lambda^2$  area because they drive off-chip wires that are significantly larger than wires on the chip. This paper does not specifically address I/O ports, but the techniques developed herein are relevant and applicable to circuits with I/O ports of realistic size.

- 2) Each input has at most constant fanout.
- 3) All instances of an input appear within constant distance of each other.

These two constraints are called *constant-input-duplicates (CID)* assumptions. It is reasonable to expect that a real VLSI circuit will receive a single copy of any input, or at most a small number of copies. It is then up to the circuit to replicate the input as required and to transmit copies to distant locations on the chip as required. The following gives an example of the energy costs of replicating and transmitting input. The example shows that since these costs can have an asymptotic effect on the total energy, they cannot be neglected in practice. The constant-input-duplicates assumptions thus force the circuit designer to account for the cost of replicating and transmitting input.

Let  $A = (a_0, \dots, a_{n-1})$  and  $B = (b_0, \dots, b_{n-1})$  be sets of boolean variables. Consider the following DNF expression:

$$H(A, B) = (a_{n-1} \wedge b_{n-1}) \vee [(a_{n-1} \oplus b_{n-1}) \wedge (a_{n-2} \wedge b_{n-2})] \vee \dots \vee [(a_{n-1} \oplus b_{n-1}) \wedge \dots \wedge (a_1 \oplus b_1) \wedge (a_0 \wedge b_0)]$$

where  $a_i, b_i \in \{0,1\}, 0 \leq i < n$

$$\text{Let } \hat{A} = \sum_{i=0}^{n-1} a_i 2^i \text{ and } \hat{B} = \sum_{i=0}^{n-1} b_i 2^i$$

$H(A, B)$  computes the  $n+1$ st bit of  $\hat{A} + \hat{B}$ .

Consider a circuit  $C_n$  for  $H$  in which each occurrence of  $a_i$  and  $b_i$  in  $H(A, B)$  is provided at a distinct input node, and interior nodes have fanout at most one. Then  $C_n$  is a tree with  $O(n^2)$  leaves. Brent and Kung [BK80] and Yao [Ya81] showed that such a tree requires area  $\Omega(n^2 \log n)$  when it has  $O(\log n)$  depth and the leaves are on a convex boundary of the layout. Thus, an optimal embedding of  $C_n$  in *USM* will use no more than  $O(n^2 \log n)$  energy, which is consumed when  $H$  switches from a state where  $A = B = 0^n$  to a state where  $A = 1^n$  and  $B = 0^{n-1}1$ .

## Upper and Lower Bounds on Switching Energy in VLSI

The energy cost of  $H$  can be improved to  $O(n^2)$  by using a nontree-like circuit  $D_n$  in which the conjunctive clauses of  $H$  are energy-efficient (ie. use at most  $O(n)$  energy per clause). The technique for realizing such a circuit is discussed in section 4. As in  $C_n$ , each occurrence of  $a_i$  and  $b_i$  in  $H(A, B)$  is provided at a distinct input node in  $D_n$ .

The analysis above charges only unit energy per input instance, although many inputs have  $O(n)$  instances and instances can be up to  $O(n^2)$  distance apart when the input nodes are laid out as in Figure 2.1. Consider an analysis of  $H$  that realistically accounts for these factors.

Figure 2.1 illustrates a circuit  $\hat{D}_n$  for  $H$  that includes the input fanout trees. The example in the Figure uses  $n = 4$ . Recall that F denotes a fanout node that computes the identity function of its input. Consider the area of  $\hat{D}_n$ .

More than half the  $2n$  inputs each have at least  $n/2$  instances in  $H$ . Hence the area of each of these input fanout trees in  $\hat{D}_n$  is  $\Omega(n \log n)$ , even when the large separation between instances is ignored. When an input switches the entire fanout tree switches. Hence the total energy for the input fanout trees is  $\Omega(n^2 \log n)$ , which exceeds the  $O(n^2)$  energy cost of the non-input portion of  $\hat{D}_n$  (assuming  $\hat{D}_n$  uses energy-efficient conjunctive clauses).

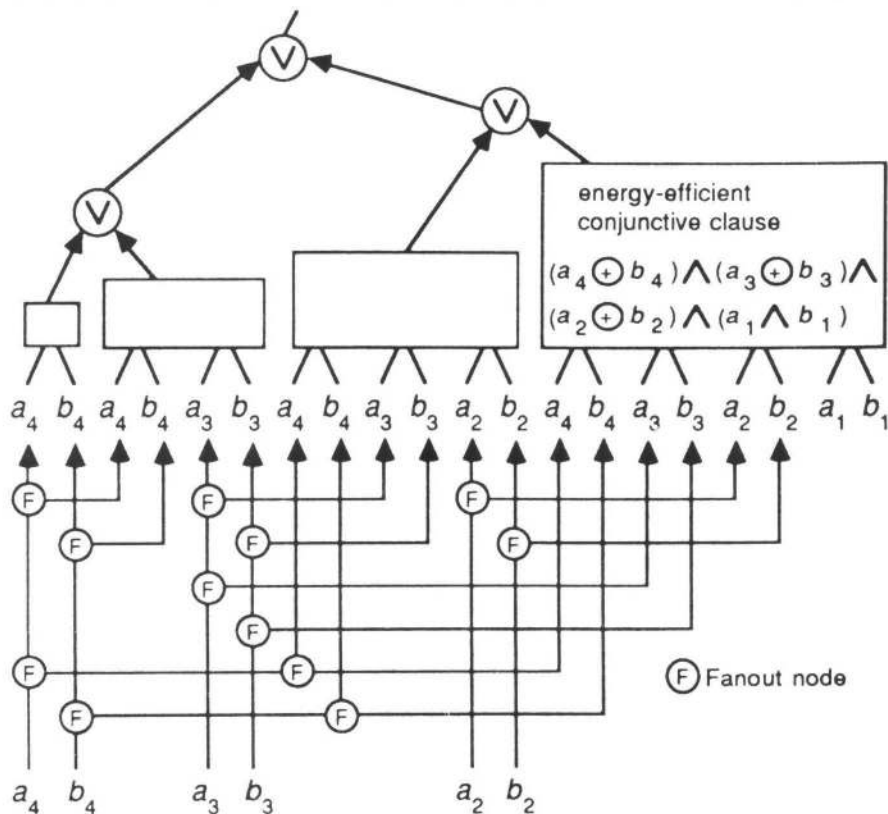


Figure 2.1.  $\hat{D}_4$ , a VLSI circuit with large input fanout

## Upper and Lower Bounds on Switching Energy in VLSI

If we realistically assume that an input will arrive at a single input port, then the large separation between input instances in  $D_n$  will be manifested by long fanout wires in  $\hat{D}_n$ . In fact,  $\Omega(n)$  inputs in  $\hat{D}_n$  have instances that are  $\Omega(n^2)$  apart. This drives the energy cost to at least  $\Omega(n^3)$ .

In chapter 4,  $H$  is shown to be computable in  $O(n)$  energy in  $USM$ .

### 3. LOWER BOUNDS

#### 3.1 Trivial Bounds

The trivial lower bound for the worst case switching energy of a circuit with  $n$  inputs is  $\Omega(n)$  (for cyclic and acyclic circuits), achieved when all inputs are switched. In  $USM$ , an acyclic circuit of area  $A$  uses  $E_{worst} = O(A)$  trivially. Hence, in cases where  $A = O(n)$  then  $E_{worst} = \Theta(n)$ . For many  $n$ -input functions this is achieved with circuits of  $O(n)$  depth. For example, an  $n$ -stage ripple carry adder, which has depth  $O(n)$ , uses  $E_{worst} = \Theta(n)$  since area  $A = O(n)$ . Thus, in order to obtain superlinear lower bounds for energy, most of the theorems in this section assume sublinear circuit depth.

However, assuming sublinear depth is not always sufficient to guarantee a circuit of superlinear area. For example, the well known H-tree embedding, illustrated in Figure 3.1, can be used to realize a parity function on  $n$  inputs, where the nodes of the tree are exclusive-or ( $\oplus$ ) gates. Such a circuit has *shallow* (ie.  $\log n$ ) depth but only  $O(n)$  area and hence  $\Theta(n)$  energy in  $USM$ . However, the H-tree embedding has its input ports strewn throughout the layout, while in practice it is advisable to confine I/O ports to convex boundaries of the layout [Me80].

**Theorem 3.0:** ( Brent, R.P. & H.T. Kung [BK80], A. Yao [Ya81] )

A tree with  $n$  leaves on a convex boundary and of depth  $D$  requires area  $A \geq \frac{cn \log n}{\log(2D/\log n)}$  for  $c > 0$ .

In particular, if  $D = O(\log n)$  then  $A = \Omega(n \log n)$ . Thus, to obtain superlinear energy bounds, many of the results in this section assume that  $n$ -input circuits have sublinear depth, and the input (and output) ports are on a convex boundary of the layout. In many cases, Theorem 3.0 above thus guarantees that the embedded circuits have superlinear area.

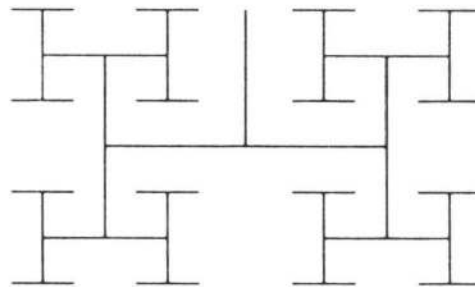


Figure 3.1 H-tree Layout

### 3.2 Monotone Circuits

**Definition:** [Sa76]

A *monotone circuit* is a circuit whose noninput nodes are labeled with functions from the monotone basis  $\{\wedge, \vee\}$ .

**Theorem 3.1:**

A monotone circuit  $C_n$  without constant inputs, embedded in area  $A$  requires worst case energy  $E_{worst}(C_n) = \Omega(A)$ .

**Proof:**

Assume  $C_n$  is in a legal state. Thus  $C_n$  may contain some "0" edges and some "1" edges. Let  $A_0$  be the area of the "0" edges. Let  $A_1$  be the area of the "1" edges.

case 1:

If  $A_0 \geq \frac{1}{2} * A$  then apply  $1^n$  to the input nodes. This will force all "0" edges to switch. Therefore,  
 $E_{worst}(C_n) = A_0 \geq \frac{1}{2} * A$ .

case 2:

If  $A_0 < \frac{1}{2} * A$  then apply  $0^n$  to the input nodes. This will force all "1" edges to switch. Therefore  
 $E_{worst}(C_n) = A_1 \geq \frac{1}{2} * A$ .

[]

Theorem 3.1 shows that in *USM*, a monotone circuit will switch most of its area in the worst case. Thus, the naive way of realizing the *OR* function on  $n$  inputs, with a monotone tree of  $\vee$ -gates, uses worst case energy proportional to the area of the tree. This high energy expenditure can be reduced for *OR*( $n$ ) and other functions by introducing negations into the circuit and by using a novel layout. Section 4 describes such a VLSI circuit,  $C_n$ , for computing *OR*( $n$ ), which uses a complete basis.  $C_n$  has  $O(\log n)$  depth,  $O(n \log n)$  area, but uses only  $O(n)$  worst case uniswitch energy, which is at least a log factor better than a shallow depth monotone circuit for *OR*( $n$ ).

### 3.3 Multiple Output Functions

The previous section gave a general energy lower bound for monotone circuits. No such nontrivial bound exists for the class of circuits over a complete basis. In this section, a class  $L$  of  $n$ -input functions is defined for which superlinear energy is required if the functions are realized by a circuit of sublinear depth, when the I/O ports are on a convex boundary of the layout. Class  $L$  includes addition and multiplication of two  $n$ -bit binary numbers and other common multiple output functions.

Intuitively, each  $n$ -input function in  $L$  is shown to have the property that many (ie.  $\Omega(n)$ ) outputs can be switched by switching only 1 input. Hence these functions are called *1-switchable*. 1-switchability is shown to imply the existence of many switched paths between the switched outputs and the single switched input. By observing that these paths require large (ie.  $\Omega(n \log n)$ ) area, a large energy bound is obtained. The following discussion formalizes these notions.

If  $\alpha$  and  $\beta$  are two boolean bit strings of length greater than zero, then  $H(\alpha, \beta) = 1$  denotes that  $\alpha$  and  $\beta$  differ in only one bit.  $H(\alpha, \beta)$  is called the *Hamming distance* of the two strings.

**Definition:**

Let  $f^{(n)} = (f_1, \dots, f_{m(n)}) : \{0,1\}^n \rightarrow \{0,1\}^{m(n)}$ . If  $\forall n \exists \alpha_n, \beta_n \in \{0,1\}^n \ni H(\alpha_n, \beta_n) = 1$ , and

## Upper and Lower Bounds on Switching Energy in VLSI

$S_n = \{ i : \text{for } 1 \leq i \leq m, f_i(\alpha_n) \neq f_i(\beta_n) \}$ ; then if  $|S_n| = \Omega(n)$ , then  $f = (f^{(n)})_{n \in \mathbb{N}}$  is 1-switchable.

### Lemma 3.1:

Let  $C_n$  be a circuit with  $n$  inputs  $(x_1, x_2, \dots, x_n)$  and let  $z_j$  be a node of  $C_n$ . Let  $s_1, s_2$  be two states of circuit  $C_n$  such that  $\exists i \ni 1 \leq i \leq n$  and  $s_1(x_i) \neq s_2(x_i)$  and  $\forall k \ni 1 \leq k \leq n$  and  $k \neq i$ ,  $s_1(x_k) = s_2(x_k)$ ; and  $s_1(z_j) \neq s_2(z_j)$ . When  $C_n$  switches between states  $s_1$  and  $s_2$ , then a path in  $C_n$  from  $x_i$  to  $z_j$  switches.

### Proof:

#### case 1:

$z_j$  is node  $x_i$  for some  $i \ni 1 \leq i \leq n$ . Done.

#### case 2:

$z_j$  is a noninput node. Then at least one of the input edges to  $z_j$  switches. Call this switched edge  $e_j$ . Then the node at the tail of  $e_j$  switches. Continue on in this way. Since  $C_n$  is acyclic and  $x_i$  is the only input that switches, this process yields a switched path from  $x_i$  to  $z_j$ . []

### Theorem 3.2:

Let  $d$  be an integer  $\geq \lceil \log(n+1) \rceil$ . If a boolean  $n$ -input function  $f_n$  is 1-switchable, then to compute  $f_n$  with a VLSI circuit  $C_n$  of depth  $d$  where the I/O ports are on a convex boundary of the layout requires  $E_{worst}(C_n) = \Omega\left(\frac{n \log n}{\log(2d/\log n)}\right)$ .

### Proof:

Consider VLSI circuit  $C_n$  that realizes function  $f_n$ . By hypothesis,  $f_n$  is 1-switchable.  $\therefore \exists$  two states of  $C_n$ ,  $s_1$  and  $s_2$  in which only one input, say  $x_i$  has a different value. And,  $\exists$  set  $S$  of outputs such that  $|S| = \Omega(n)$  and  $\forall z \in S$ ,  $z$  switches when  $C_n$  switches between  $s_1$  and  $s_2$ . By Lemma 3.1,  $\exists$  a switched path from  $x_i$  to each member of  $S$  when  $C_n$  switches between  $s_1$  and  $s_2$ . By the definition of USM,  $C_n$  contains at most a constant  $k$  number of instances of input  $x_i$ . Hence, at least one instance of  $x_i$  must account for  $m \geq \frac{|S|}{k} = \Omega(n)$  switched paths between  $x_i$  and elements of  $S$ . These switched paths form a tree with  $m$  leaves on a convex boundary and depth  $d$ , which requires area  $\Omega\left(\frac{n \log n}{\log(2d/\log n)}\right)$  by Theorem 3.0. []

Note, in particular, that if  $d = O(\log n)$  then  $E_{worst}(C_n) = \Omega(n \log n)$ .

Consider the following problem list  $L$  :

#### 1) Integer Addition

input  $(x_0, \dots, x_{p-1}, y_0, \dots, y_{p-1})$  where  $x_i, y_i \in \{0,1\}$

output  $(z_0, \dots, z_p)$  where  $z_i \in \{0,1\}$  and

$$X = \sum_{i \geq 0} x_i 2^i, \quad Y = \sum_{i \geq 0} y_i 2^i, \quad Z = X + Y = \sum_{i \geq 0} z_i 2^i$$

$$n = 2p$$

$$|Z| = p + 1 = \frac{n}{2} + 1$$

#### 2) Cyclic Shift

input  $(x_0, \dots, x_{p-1}, s)$  where  $x_i \in \{0,1\}$ ,  $0 \leq s < p$

output  $(z_0, \dots, z_{p-1})$  where  $z_i = x_{(i+s) \bmod p}$

$$n = p + \lceil \log p \rceil$$

$$|Z| = p = O(n)$$

## Upper and Lower Bounds on Switching Energy in VLSI

### 3) Integer Multiplication

input  $(x_0, \dots, x_{p-1}, y_0, \dots, y_{p-1})$  where  $x_i, y_i \in \{0,1\}$

output  $(z_0, \dots, z_{2p-1})$  where  $z_i \in \{0,1\}$  and

$$X = \sum_{i \geq 0} x_i 2^i, \quad Y = \sum_{i \geq 0} y_i 2^i, \quad Z = X * Y = \sum_{i \geq 0} z_i 2^i$$

$$n = 2p$$

$$|Z| = 2p = n$$

### 4) Product of 3 Matrices over $\mathbf{Z}_2$

input  $(x_{11}, \dots, x_{pp}, y_{11}, \dots, y_{pp}, w_{11}, \dots, w_{pp})$  where  $x_{ij}, y_{ij}, w_{ij} \in \{0,1\}$

output  $(z_{11}, \dots, z_{pp})$  where

$$z_{ij} = \sum_k (v_{ik} * w_{kj}) \bmod 2 \quad \text{and} \quad v_{ik} = \sum_l (x_{il} * y_{lk}) \bmod 2$$

$$n = 3p^2$$

$$|Z| = p^2 = \frac{n}{3}$$

### 5) Binary-to-Unary

input  $(x_0, \dots, x_{(\log p)-1})$  where  $x_i \in \{0,1\}$

output  $(z_0, \dots, z_{p-1})$  where  $X = \sum_{i \geq 0} x_i 2^i$  and  $z_i = \begin{cases} 1 & \text{if } i \leq X \\ 0 & \text{if } i > X \end{cases}$

$$n = \log p$$

$$|Z| = p = 2^n$$

### Theorem 3.3:

The functions described in problem list  $L$  are 1-switchable.

#### Proof:

For each function  $f$  described in  $L$ , two states,  $s_1, s_2$  are given below by defining the input configurations  $(X, Y, W)$  and the output configuration  $Z$ .  $p$  is defined in  $L$  for each function. The reader can verify that for each problem in  $L$ , when a circuit for  $f$  with  $n$  inputs switches between  $s_1$  and  $s_2$ , then one input bit switches and  $\Omega(n)$  output bits switch.

#### 1) Integer Addition

$$s_1: X = 1^p, Y = 0^p \Rightarrow Z = 01^p$$

$$s_2: X = 1^p, Y = 0^{p-1}1 \Rightarrow Z = 10^p$$

#### 2) Cyclic Shift

for  $p$  even:

$$s_1: X = ((10)^{\frac{p}{2}}, 0) \Rightarrow Z = (10)^{\frac{p}{2}}$$

$$s_2: X = ((10)^{\frac{p}{2}}, 1) \Rightarrow Z = (01)^{\frac{p}{2}}$$

for  $p$  odd:

$$s_1: X = ((10)^{\lfloor \frac{p}{2} \rfloor} \left\lfloor \frac{p}{2} \right\rfloor 1, 0) \Rightarrow Z = (10)^{\lfloor \frac{p}{2} \rfloor} \left\lfloor \frac{p}{2} \right\rfloor 1$$

$$s_2: X = ((10)^{\lfloor \frac{p}{2} \rfloor} \left\lfloor \frac{p}{2} \right\rfloor 1, 1) \Rightarrow Z = (01)^{\lfloor \frac{p}{2} \rfloor} \left\lfloor \frac{p}{2} \right\rfloor 1$$

3) Integer Multiplication

$$s_1: X = 1^p, Y = 0^p \Rightarrow Z = 0^{2p}$$

$$s_2: X = 1^p, Y = 0^{p-1}1 \Rightarrow Z = 0^p 1^p$$

4) Product of 3 Matrices over  $Z_2$

for  $p$  even:

$$s_1: [x_{ij}] = [y_{ij}] = [w_{ij}] = 1^{pp} \Rightarrow [z_{ij}] = 0^{pp}$$

$$s_2: [x_{ij}] = [w_{ij}] = 1^{pp}$$

$$[y_{ij}] = \begin{bmatrix} 0 & 1 & \dots & 1 \\ 1 & \dots & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix} \Rightarrow [z_{ij}] = 1^{pp}$$

(ie.  $y_{11}=0, y_{ij}=1 \forall ij \neq 11$ )

for  $p$  odd:

same as  $p$  even except last row of  $[y_{ij}]$  is  $0^p$  in  $s_1$  and  $s_2$ .

5) Binary-to-Unary

$$s_1: X = 0^{\log p} \Rightarrow Z = 0^p$$

$$s_2: X = 10^{(\log p)-1} \Rightarrow Z = 1^{\frac{p}{2}} 0^{\frac{p}{2}}$$

[]

**Corollary 3.1:**

Let  $d$  be an integer  $\geq \lceil \log(n+1) \rceil$ . The functions in  $L$  require  $E_{worst} = \Omega\left(\frac{n \log n}{\log(2d/\log n)}\right)$  to be computed by a VLSI circuit of depth  $d$  where the I/O ports are on a convex boundary of the layout.

**Proof:**

By Theorem 3.3, the functions in  $L$  are 1-switchable.  $\therefore$  by Theorem 3.2, the functions of  $L$  require  $E_{worst} = \Omega\left(\frac{n \log n}{\log(2d/\log n)}\right)$  []

**3.3.1 Related Work**

Several researchers have studied a subclass of 1-switchable functions called *transitive* functions, which includes integer multiplication and matrix multiplication. The set of transitive functions was defined by Vuillemin [Vu83], and Snyder and Tyagi [ST86] showed that the transitive functions form a proper subset of the 1-switchable functions.

Lengauer and Mehlhorn [LM81] obtained an  $\Omega(n + (n^2/\log(A/n^2)))$  bound on the uniswitch energy of transitive functions. Snyder and Tyagi [ST86] rederived this result in the case where  $A = O(n^2)$ . The bounds obtained by both [LM81] and [ST86] use information theoretic arguments that preclude encodings. Aggarwal et al [AGR88] improved their result to obtain an  $\Omega(n^2)$  worst case bound



## Upper and Lower Bounds on Switching Energy in VLSI

on the uniswitch energy of transitive functions.

[AGR88] also showed that if the I/O ports of an adder need not be on the periphery of the layout, then addition can be computed in  $O(n \log n / (\log \log n))$  uniswitch energy. Section 5 of this paper shows that addition can be computed in *linear* average energy while keeping the I/O ports on the periphery of the layout.

Snyder and Tyagi [ST86] have extended the result on 1-switchable functions to a range of depths. In particular, they showed that a convex VLSI circuit  $C$  that computes a 1-switchable function in depth  $d(n)$ ,  $\log^2 n \leq d(n) \leq n^\epsilon$ ,  $0 < \epsilon \leq 1$ , requires  $E_{worst}(C) * d(n) = \Omega(\max(n \log n, nd(n)))$ .

### 3.4 Single Output Functions

The proof techniques of the previous section are applicable only to multi-valued functions. Section 4 describes a method for obtaining VLSI circuits for certain  $n$ -input predicates (ie. single-valued functions), which use  $E_{worst} = O(n)$ . These predicates include *OR* and *AND* functions on  $n$  inputs, and compare functions. However, it is unlikely that all  $n$ -input predicates that can be computed by a VLSI circuit of shallow (ie.  $O(\log n)$ ) depth can be computed in  $O(n)$  worst case energy. The following discussion provides evidence for this conjecture, by describing a superlinear lower bound on parity, for a specialized basis.

Consider the *parity* function on  $n$  boolean variables (ie.  $x_1 \oplus x_2 \oplus \dots \oplus x_n$ ). In the special case where the circuit basis is  $\{\oplus, \neg\}$ , a superlinear energy lower bound for parity is derived in the following theorem, which was independently obtained by J. Leo [Le84].

#### Theorem 3.4:

To compute parity of  $n$  inputs with a VLSI circuit  $C_n$  of area  $A$  requires  $E_a(C_n) = \Omega(A)$  when the circuit basis is  $\{\oplus, \neg\}$  and when  $C_n$  contains no constant inputs and no nodes that compute a constant function.

#### Proof:

Let  $W = \{w\}$  be the wires of  $C_n$ .

Note that when the basis for  $C_n$  is  $\{\oplus, \neg\}$ , each node of  $C_n$  computes a parity function of a nonempty subset of the inputs or their negations.

The inputs of  $C_n$  are assumed to be uniformly distributed over  $\{0,1\}$  (by the definition of  $E_a$ ).

Hence, each wire of  $C_n$  has value 1 (or 0) for exactly half the states of  $C_n$ .

$\therefore \forall w \in W, \Pr(w \text{ switches}) = 1/2$ .

$\therefore E_a(C_n) \geq \frac{1}{2} A$ .  $\square$

The definitions of  $E_a$  and  $E_{worst}$  yield the following Corollary to Theorem 3.4.

#### Corollary 3.2:

To compute parity of  $n$  nonconstant inputs with a VLSI circuit  $C$  of area  $A$  requires  $E_{worst}(C) = \Omega(A)$  when the circuit basis is  $\{\oplus, \neg\}$ .

An alternate proof of the *worst case* lower bound for parity is presented below in Theorem 3.4A. The alternate proof yields a deterministic polynomial algorithm for computing a pair of states that induces a lot of energy. Theorems 3.4 and 3.4A together demonstrate the relative difficulty of average case analysis versus worst case analysis.

**Corollary 3.3:**

To compute the parity function on  $n$  boolean variables with a VLSI circuit  $C_n$  of  $O(\log n)$  depth with the I/O ports on a convex boundary of  $C_n$  requires  $E_a(C_n) = \Omega(n \log n) = E_{worst}(C_n)$  when the basis for  $C_n$  is  $\{\oplus, \neg\}$ .

**Proof:**

Let  $A$  be the area of  $C_n$ . By Theorem 3.4 and Corollary 3.2,  $E_a(C_n) = \Omega(A) = E_{worst}(C_n)$ . Since circuit  $C_n$  must fanin the  $n$  inputs, and since nodes have indegree  $\leq 2$ ,  $A$  is at least as large as the area of a binary tree on  $n$  leaves. [BK80] and [Ya81] showed that such a tree requires area  $\Omega(n \log n)$  when the depth is  $O(\log n)$  and the leaves are on a convex boundary.

$$\therefore E_{worst}(C_n) \geq E_a(C_n) \geq \frac{1}{2}A = \Omega(n \log n). \quad []$$

Theorem 3.4A below provides a direct proof of the worst case lower bound for parity, in the special case where the circuit basis is  $\{\oplus\}$ . The reader will note the relative complexity of the deterministic proof technique used in Theorem 3.4A, compared to the simple probabilistic argument used to prove the stronger result of Theorem 3.4. Theorem 3.4A is primarily due to Stephen Cook and uses an observation of Leslie Valiant [Va84].

**Theorem 3.4A:**

To compute parity on  $n$  boolean variables with a VLSI circuit  $C_n$  of area  $A$  and  $O(\log n)$  depth requires  $E_{worst}(C_n) = \Omega(A)$  when the basis for  $C_n$  is  $\{\oplus\}$  and when  $C_n$  contains no constant inputs and no nodes that compute a constant function.

**Proof:**

Note that when the basis for  $C_n$  is  $\{\oplus\}$ , each node of  $C_n$  computes a parity function of a nonempty subset of the inputs. Let  $S$  be the set of nodes of  $C_n$ . Let  $X = (x_1, \dots, x_n)$  be the input nodes (variables) of the circuit (function).

**Definition:**

Let  $p \in S$  and let  $f_p$  be the parity function computed at node  $p$ . Let  $X_p \subseteq X \ni f_p$  is the parity function of inputs  $X_p$ . Let  $w_{p1}, w_{p2}$  denote the output edges from  $p$ . (Recall that  $C_n$  has fanout  $\leq 2$ ). Let  $weight(p) = area(w_{p1}) + area(w_{p2})$ . When  $p$  has fanout 1,  $area(w_{p2}) = 0$ . When  $p$  has fanout 0,  $weight(p) = 0$ .

**Lemma 3.2:** [Stephen Cook]

There exists an assignment  $B$  of boolean values to  $x_1, \dots, x_n$  such that when  $C_n$  is in state  $B$ , then

$$\sum_{p \in S \wedge (f_p(x_1, \dots, x_n) = 1)} weight(p) \geq \frac{1}{2}A$$

**Proof of Lemma 3.2:**

The following construction sequentially defines an assignment  $B$  of values to the inputs  $- B(x_1), \dots, B(x_n) -$  that will cause at least half the area of  $C_n$  to be "1". In the following,  $S_k$  is the subset of nodes of  $C_n$  that depends only on the inputs  $x_1, \dots, x_k$ .  $A_k$  is the area of the out edges of nodes in  $S_k$ .

More formally,

let  $S_k = \{ p \in S : x_k \in X_p \text{ and } \forall i > k, x_i \notin X_p \}$  and

let  $A_k = \sum_{p \in S_k} weight(p)$

Basis of assignment:  $B(x_1) = 1$

$$B(x_1) = 1 \Rightarrow \sum_{p \in S_1 \wedge (f_p(x_1) = 1)} weight(p) = A_1$$

## Upper and Lower Bounds on Switching Energy in VLSI

In general, suppose  $B(x_1), \dots, B(x_{k-1})$  have been determined. To determine  $B(x_k)$ :

There are two choices for  $B(x_k)$ :

Suppose  $B(x_k) = 0$ .

$$\text{Let } W_0 = \sum_{p \in S_k \wedge (f_p(X_p)=0)} \text{weight}(p)$$

$$\text{Let } W_1 = \sum_{p \in S_k \wedge (f_p(X_p)=1)} \text{weight}(p)$$

case 1:

If  $W_1 \geq \frac{1}{2}A_k$  then done. ie.  $B(x_k) = 0$

case 2:

If  $W_1 < \frac{1}{2}A_k$  then set  $B(x_k) = 1$ . Since  $\forall p \in S_k, f_p$  is a parity function and  $x_k \in X_p$ ; then it follows that changing  $B(x_k)$  from 0 to 1 changes  $f_p(X_p)$  from 0 to 1.

Note that setting  $x_k$  does not affect the functions realized by nodes in  $S_i$  for  $1 \leq i < k$ .

$$\therefore \sum_{p \in S_k \wedge (f_p(X_p)=1)} \text{weight}(p) \geq \frac{1}{2}A_k$$

$$\therefore \sum_{p \in S \wedge (f_p(X_p)=1)} \text{weight}(p) \geq A_1 + \frac{1}{2} \sum_{k=2}^n A_k \geq \frac{1}{2}A$$

[] (end of Lemma 3.2)

Since  $C_n$  consists of  $\oplus$  nodes only,  $X = 0^n \Rightarrow$  all wires in  $C_n$  have value 0. Let  $B(X)$  be the value of  $X$  determined by Lemma 3.2. If  $C_n$  is switched such that  $X : 0^n \rightarrow B(X)$ , then  $E_{worst}(C_n) \geq \frac{1}{2}A$ .

[] (end of Theorem 3.4A)

### 3.5. Open Problems

The USM lower bounds on parity are derived in the special case where the circuit contains only  $\oplus$ -gates and negations.

#### Conjecture:

To compute the parity function on  $n$  bits by an  $O(\log n)$  depth circuit in which the inputs are on a convex boundary requires  $\Omega(n \log n)$  uniswitch energy.

The conjecture above does not restrict the basis of the parity circuit. Note that in order to obtain an  $\Omega(\text{area})$  uniswitch lower bound for parity in the general case, a notion of a "minimal" circuit is required. This is because an extraneous circuit that uses  $o(\text{area})$  energy can always be "attached" to a parity circuit.

What about the *majority* function? We believe that majority also requires superlinear uniswitch energy if computed by a shallow depth circuit.

## 4. WORST CASE UPPER BOUNDS

### 4.1 Energy-Efficient OR and AND Circuits

The energy-efficient *OR* circuit described in this section evolved from the simple observation that it is sufficient to turn on one *OR* input to turn on the output. Therefore, intuitively, even when many or all the inputs are turned on, only one of the "1" signals need propagate all the way to the output. In a completely analogous manner, it is sufficient to turn off one *AND* input in order to turn off the output. Thus,

when many inputs are turned off, only one "0" signal must propagate all the way to the output. This is the essence of the *SOR* (*S* mart *OR*) circuit and the *SAND* (*S* mart *AND*) circuit, described below. When many inputs to *SOR* are "1", all but one of these "1" signals are "killed", using the dual of *SOR*, which is *SAND*. Similarly, extraneous *SAND* inputs are "killed" using *SOR* signals.

The layout of the *SOR/SAND* circuit is designed so that the area used to "kill" signals (ie. prevent "1" inputs from reaching the *SOR* output, and prevent "0" inputs from reaching the *SAND* output) is at most linear in the input size, and the area of both the "successful" path to an output plus the "killed" paths is at most linear in the input size.

The following recurrences describe the boolean functions  $OR : \{0,1\}^n \rightarrow \{0,1\}$  and  $AND : \{0,1\}^n \rightarrow \{0,1\}$  in a novel way. The reader can verify that  $OR(x_1, \dots, x_n) \equiv x_1 \vee x_2 \vee \dots \vee x_n$  and  $AND(x_1, \dots, x_n) \equiv x_1 \wedge x_2 \wedge \dots \wedge x_n$ . The *USM* circuit realization of *OR* and *AND* is the energy-efficient *SOR/SAND* circuit.

**Recurrences:**

$$1) \quad OR(x_i, x_j) = x_i \vee (\bar{x}_i \wedge x_j)$$

$$OR(x_1, \dots, x_n) = OR(x_1, \dots, x_{n/2}) \vee [AND(\bar{x}_1, \dots, \bar{x}_{n/2}) \wedge OR(x_{(n/2)+1}, \dots, x_n)]$$

$$2) \quad AND(x_i, x_j) = (x_i \vee \bar{x}_j) \wedge x_j$$

$$AND(x_1, \dots, x_n) = [AND(x_1, \dots, x_{n/2}) \vee OR(\bar{x}_{(n/2)+1}, \dots, \bar{x}_n)] \wedge AND(x_{(n/2)+1}, \dots, x_n)$$

$OR(x_1, \dots, x_n)$  is abbreviated by  $OR(n)$ .  $OR(\bar{x}_1, \dots, \bar{x}_n)$  is abbreviated by  $\overline{OR}(n)$ . *AND* is similarly abbreviated.  $(x_1, \dots, x_n)$  is also written as  $(x_i; x_n)$ .

The discussion that follows is a formal description of the construction used to obtain energy-efficient VLSI circuits. To clarify the formalism, the reader is advised to refer to Figures 4.0 and 4.1, which illustrate a VLSI circuit called *LF*. *LF* is an embedding in the plane of circuit *SOR/SAND*, which computes the functions *OR/AND*. Figure 4.0 illustrates *LF* on 2 inputs. Figure 4.1 recursively depicts *LF* on *n* inputs. Circuit *SOR/SAND* and layout *LF* are precisely defined below.

**Definition:**

$SOR/SAND(n) = (V_{SD}, W_{SD})$  is a circuit, illustrated in Figures 4.0 and 4.1, such that

$V_{SD} = I_1 \cup I_2 \cup L$  where

$I_1$  are the input nodes  $\{x_1, x_2, \dots, x_n\}$  and

$I_2$  are the input nodes  $\{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$ .

$L$ , the set of interior nodes, is as follows.

$$L = \{(\vee, v_1^{i,k}), (\vee, v_2^{i,k}), (\wedge, v_3^{i,k}), (\wedge, v_4^{i,k}) \text{ where } 1 \leq i \leq \log_2 n, 1 \leq k \leq \frac{n}{2^i} \}$$

$\{v_1^{\log n, 1}, v_4^{\log n, 1}\}$  are the output nodes. For consistency,  $x_k$  is also denoted  $v_1^{0,k}$  and  $\bar{x}_k$  is also denoted as  $v_4^{0,k}$ . The nodes of  $I_1$  and  $I_2$  are labeled to indicate that the inputs that occur at nodes of  $I_2$  are the negation of those at nodes of  $I_1$ .

$W_{SD}$ , the set of edges, is as follows.

## Upper and Lower Bounds on Switching Energy in VLSI

$$W_{SD} = \{e_j^{i,k} \mid 1 \leq j \leq 8, 1 \leq i \leq \log_2 n, 1 \leq k \leq \frac{n}{2^i} \text{ and}$$

$$e_1^{i,k} = (v_1^{i-1, 2k-1}, v_1^{i,k}), e_2^{i,k} = (v_4^{i-1, 2k-1}, v_2^{i,k}),$$

$$e_3^{i,k} = (v_4^{i-1, 2k-1}, v_3^{i,k}), e_4^{i,k} = (v_1^{i-1, 2k}, v_3^{i,k}),$$

$$e_5^{i,k} = (v_1^{i-1, 2k}, v_2^{i,k}), e_6^{i,k} = (v_4^{i-1, 2k}, v_4^{i,k}),$$

$$e_7^{i,k} = (v_3^{i,k}, v_1^{i,k}), e_8^{i,k} = (v_2^{i,k}, v_4^{i,k})\}$$

The indices  $i$ ,  $j$  and  $k$  are used to label the nodes and edges of *SOR/SAND* uniquely. The subscript  $j$  distinguishes between types of nodes and edges, and superscripts  $i$  and  $k$  distinguish within a type. In particular,  $i$  indexes *SOR/SAND* along a vertical axis, increasing from 0 at the inputs along the bottom to  $\log n$  (ie.  $\text{depth}(\text{SOR/SAND})/2$ ) at the top.  $i$  is thus called a *vertical index*.  $k$  indexes *SOR/SAND* along a horizontal axis, increasing from left to right, and is called a *horizontal index*.

Recall from section 2 that  $s$  is a state function that attributes boolean values to the nodes and wires of a circuit. Thus, for  $v \in V_{SD}$ ,  $s(v)$  denotes the value of node  $v$ . In the upcoming analysis, when the input is not clear from context,  $s_X(v)$  will denote the value of node  $v$ , where  $X = (x_1, \dots, x_n)$  is the input to *SOR/SAND* ( $n$ ). Alternately, for  $i \in \mathbf{N}$ ,  $s_i(v)$  denotes the value of node  $v$  at time  $t_i$ . Similarly, for  $w \in W_{SD}$ ,  $s(w)$  (or  $s_X(w)$  or  $s_i(w)$ ) denotes the value of the node at the tail of wire  $w$ . The state function is extended to sets of nodes and wires as follows. For  $U \subseteq V_{SD} \cup W_{SD}$ ,  $s(U) = \{s(u) : u \in U\}$ .

Let  $F_n$  be the function realized by circuit *SOR/SAND* ( $n$ ).

$$F_n : \{0,1\}^n \rightarrow \{0,1\}^2 \text{ such that } F_n(X) = (v \uparrow^{\log n, 1}, v \downarrow^{\log n, 1})$$

The reader can verify that

$$v \uparrow^{\log n, 1} = \text{OR}(x_1, \dots, x_n) \text{ and}$$

$$v \downarrow^{\log n, 1} = \overline{\text{OR}}(x_1, \dots, x_n) = \text{AND}(\bar{x}_1, \dots, \bar{x}_n)$$

Layouts of *SOR/SAND* (2) and *SOR/SAND* ( $n$ ) are illustrated, respectively, in Figures 4.0 and 4.1.  $F_n(x_1, \dots, x_n)$  is abbreviated as  $F_n(X)$  or  $F(n)$ . The following defines an embedding of a circuit in the plane.

### Definition:

An  $(I, J)$ -grid-with-diagonals  $GD_{IJ} = (\hat{V}, \hat{E})$  is a graph where  $\hat{V} = \{(k, m) \mid 0 \leq k < I, 0 \leq m < J\}$  (ie. set of cartesian coordinates), and edges of  $\hat{E}$  join vertex pairs that are either unit distance apart or distance  $\sqrt{2}$  apart.  $GD_{44}$  is illustrated in Figure 4.2.

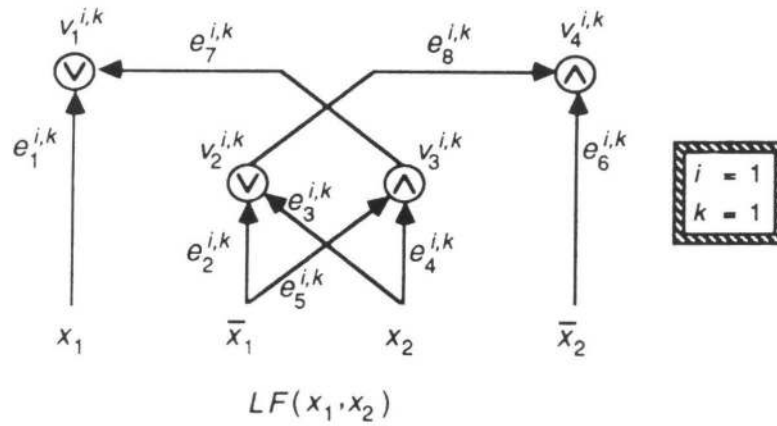
### Definition:

A layout (embedding, placement),  $\Psi$ , of graph  $G = (V, E)$  into  $GD_{IJ}$  is a 1-to-1 mapping of  $V$  into  $\hat{V}$  and  $E$  into paths (wires) of  $\hat{E}$  such that  $\forall (x, y) \in E$ ,  $\Psi(x, y)$  is a path from  $\Psi(x)$  to  $\Psi(y)$ , and every pair of paths in  $\hat{E}$  is edge-disjoint.

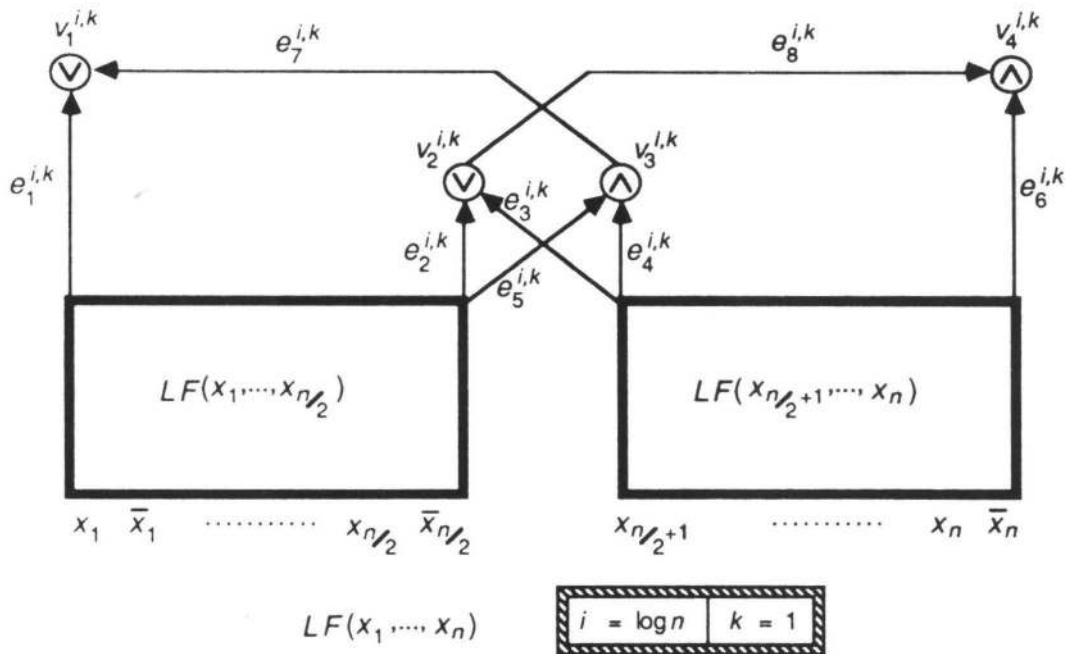
### Definitions:

$$\text{height}(GD_{IJ}) \triangleq I-1$$

$$\text{width}(GD_{IJ}) \triangleq J-1$$



An embedding of circuit *SOR/SAND* (2)  
Figure 4.0.



An embedding of circuit *SOR/SAND* ( $n$ )  
Figure 4.1.

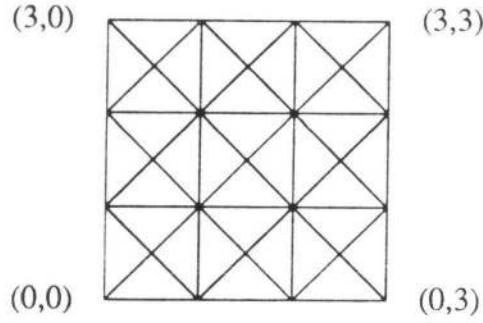


Figure 4.2.  $GD_{44}$

$$area(GD_{1j}) \triangleq height(GD_{1j}) * width(GD_{1j})$$

Let  $LF(x_1, x_2, \dots, x_n) = \Psi(SOR/SAND(n))$  such that input nodes are unit spaced on a line, with  $\bar{x}_j$  placed to the immediate right of  $x_j$  for  $1 \leq j \leq n$ , and the wire lengths are as follows.

$$\|e_{1,k}^i\| = \|e_{6,k}^i\| = 2, \|e_{2,k}^i\| = \|e_{4,k}^i\| = 1, \|e_{3,k}^i\| = \|e_{5,k}^i\| = \sqrt{2}, \text{ and}$$

$$\|e_{7,k}^i\| = \|e_{8,k}^i\| = 2^i + \sqrt{2} - 1 \text{ for } 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}$$

The relative location of the nodes of  $SOR/SAND(n)$  in the layout is evident from the recursive description of  $LF$ , illustrated in Figures 4.0 and 4.1.  $LF(x_1, \dots, x_n)$  is abbreviated by  $LF(n)$  or  $LF$ . Some facts about  $LF(n)$  and  $SOR/SAND(n)$ :

$$\begin{aligned} 1) \text{ height}(LF(n)) &= 2 + \text{height}(LF(\frac{n}{2})) \\ &= 2\log_2 n \end{aligned}$$

$$\begin{aligned} 2) \text{ area}(LF(n)) &= \text{height}(LF(n)) * \text{width}(LF(n)) \\ &= 2\log_2 n * (2n - 1) \\ &\approx 4n \log_2 n \end{aligned}$$

Note that  $T(n)$  is a complete binary tree with  $n$  leaves unit spaced on a line,

$$\text{height}(T(n)) = \log n \text{ and}$$

$$\text{width}(T(n)) = n - 1.$$

$$\therefore \text{area}(LF(n)) \approx 4 * \text{area}(T(n)).$$

3) Let  $D(n)$  be the depth of the  $SOR/SAND(n)$  circuit.

$$\begin{aligned} D(n) &= 2 + D(\frac{n}{2}) \\ &= 2\log_2 n \end{aligned}$$

4) The reader can verify from the recurrence for  $OR(n)$  and layout  $LF(n)$  that  $\forall X \in \{0,1\}^n$   $(s_X(e_{1,k}^i), s_X(e_{7,k}^i)) \neq (1,1)$  for  $1 \leq i \leq \log_2 n, 1 \leq k \leq \frac{n}{2^i}$ . Similarly,  $\forall X \in \{0,1\}^n$   $(s_X(e_{6,k}^i), s_X(e_{8,k}^i)) \neq (0,0)$  for  $1 \leq i \leq \log_2 n, 1 \leq k \leq \frac{n}{2^i}$ , from the recurrence for  $AND(n)$  or by duality.

The formal discussion that follows derives the upper bound on the worst case energy used by  $LF$ . Intuitively, the analysis below proceeds by first partitioning  $LF$  into a subgraph of "short" wires and a subgraph of "long" wires. The short wires are shown to constitute only  $O(n)$  area, and are thus eliminated from subsequent discussion. Further, since  $LF$  consists of dual  $SOR$  and  $SAND$  subcircuits, only the long wires of the  $SOR$  subcircuit are fully analyzed. The long wires are shown to occupy  $O(n \log n)$  area, but further analysis shows they use only  $O(n)$  worst case energy. The reader is advised to refer to Figure 4.1, which depicts layout  $LF$  of circuit  $SOR/SAND(n)$ , while reading the following definitions.

**Definitions:**

Recall that circuit  $SOR/SAND(n) = (I_1 \cup I_2 \cup L, W_{SD})$ .

Let  $\vee T(m, n) = (\vee V(m, n), \vee E(m, n))$  be a labeled subgraph of  $SOR/SAND(n)$  such that

$$\vee V(m, n) = I_1 \cup \{v_1^{i,k}, v_3^{i,k} \mid 1 \leq i \leq \lceil \log(n-m+1) \rceil, \left\lfloor \frac{m}{2^i} \right\rfloor \leq k \leq \left\lfloor \frac{n}{2^i} \right\rfloor\} \text{ and}$$

$$\vee E(m, n) = \{e_1^{i,k}, e_4^{i,k}, e_5^{i,k}, e_7^{i,k} \mid 1 \leq i \leq \lceil \log(n-m+1) \rceil, \left\lfloor \frac{m}{2^i} \right\rfloor \leq k \leq \left\lfloor \frac{n}{2^i} \right\rfloor\}$$

Below,  $\wedge T(m, n)$  is defined analogously to  $\vee T(m, n)$ , as can be seen from Figure 4.1.

Let  $\wedge T(m, n) = (\wedge V(m, n), \wedge E(m, n))$  be a labeled subgraph of  $SOR/SAND(n)$  such that

$$\wedge V(m, n) = I_2 \cup \{v_2^{i,k}, v_4^{i,k} \mid 1 \leq i \leq \lceil \log(n-m+1) \rceil, \left\lfloor \frac{m}{2^i} \right\rfloor \leq k \leq \left\lfloor \frac{n}{2^i} \right\rfloor\} \text{ and}$$

$$\wedge E(m, n) = \{e_2^{i,k}, e_3^{i,k}, e_6^{i,k}, e_8^{i,k} \mid 1 \leq i \leq \lceil \log(n-m+1) \rceil, \left\lfloor \frac{m}{2^i} \right\rfloor \leq k \leq \left\lfloor \frac{n}{2^i} \right\rfloor\}$$

$$\vee \text{longwires} \triangleq \{e_7^{i,k} \mid 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}\}$$

$$\vee \text{shortwires} \triangleq \{e_1^{i,k}, e_4^{i,k}, e_5^{i,k} \mid 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}\}$$

$$\wedge \text{longwires} \triangleq \{e_8^{i,k} \mid 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}\}$$

$$\wedge \text{shortwires} \triangleq \{e_2^{i,k}, e_3^{i,k}, e_6^{i,k} \mid 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}\}$$

$$\text{longwires} \triangleq \vee \text{longwires} \cup \wedge \text{longwires}$$

$$\text{shortwires} \triangleq \vee \text{shortwires} \cup \wedge \text{shortwires}$$

In the following, node and edge indices  $i$  and  $k$  are mapped to input indices  $i_1, i_2, i_3$ , and  $i_4$ .

For  $i, k > 0$ ,

$$\text{let } i_1 = (k-1)2^i + 1,$$

$$i_2 = (2k-1)2^{i-1},$$

$$i_3 = i_2 + 1,$$

$$i_4 = k2^i$$

**Definitions:**

$$\vee \text{longwires under } e_7^{i,k} \triangleq \vee E(i_1, i_2) \cap \vee \text{longwires}$$

$$\wedge \text{longwires under } e_8^{i,k} \triangleq \wedge E(i_3, i_4) \cap \wedge \text{longwires}$$

Let  $A_{\vee s}(n)$  be the area of the  $\vee \text{shortwires}$  in layout  $LF(n)$ . Let  $A_{\wedge s}(n)$  be the area of the  $\wedge \text{shortwires}$  in layout  $LF(n)$ .

**Lemma 4.1:**

$$A_{\vee s}(n) = O(n)$$

$$A_{\wedge s}(n) = O(n)$$



**Proof:**

$$A_{\vee s}(2) = 3 + \sqrt{2}$$

$$A_{\vee s}(n) = 2A_{\vee s}\left(\frac{n}{2}\right) + (3 + \sqrt{2}) \\ \leq 10n$$

$$A_{\wedge s}(n) = O(n) \text{ by symmetry of layout } LF(n). \quad []$$

Thus, since the *shortwires* can contribute at most  $O(n)$  energy, the remaining analysis considers only the *longwires*. Further, only the  $\vee$ *longwires* are discussed in detail. The  $\wedge$ *longwires* follow by duality of the circuits and symmetry of the layouts.

**Lemma 4.2:**

If  $OR(x_1, \dots, x_n) = 0$  then  $s(\vee\text{longwires}) = \{0\}$  and  $s(\wedge\text{longwires}) = \{1\}$ .

**Proof:**

In the following analysis, a node (or edge) and its respective function share the same label.

Recall that  $\vee\text{longwires} = \{e_{i,k}^{\vee} \mid 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}\}$ . The following shows that  $s(\vee\text{longwires}) = \{0\}$  when  $OR(n) = 0$ .  $n$  is a power of 2.

$$\text{Consider } V_1 = \{v_{i,k}^{\vee} \mid 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i}\}$$

Assume for the moment that  $s(V_1) = \{0\}$ .

Since  $v_{i,k}^{\vee} = e_{i,k}^{\vee} \vee e_{i,k}^{\wedge}$ , then  $s(v_{i,k}^{\vee}) = 0 \Rightarrow s(e_{i,k}^{\vee}) = 0$ . And since the head of every  $e_{i,k}^{\vee}$  edge is a  $v_{i,k}^{\vee}$  node,  $s(V_1) = \{0\} \Rightarrow s(\vee\text{longwires}) = \{0\}$ .

It is left to show that  $s(V_1) = \{0\}$ .

Induction on  $i$ , the vertical index of  $SOR/SAND(n)$ , which computes  $OR(n)$ .

**basis:**  $i = 1$

$SOR/SAND(2)$  realizes  $OR(x_1, x_2)$ , which is the function computed by node  $v_{1,1}^{\vee}$

$$s(V_1) = \{s(v_{1,1}^{\vee})\} = \{0\} \text{ by hypothesis.}$$

**induction step:**

Consider  $v_{i,k}^{\vee} \in V_1$

$$v_{i,k}^{\vee} = v_{i-1,2k-1}^{\vee} \vee (v_{i-1,2k-1}^{\wedge} \wedge v_{i-1,2k}^{\vee})$$

By the induction hypothesis,  $s(v_{i-1,2k-1}^{\vee}) = 0$  and  $s(v_{i-1,2k}^{\vee}) = 0$ .

$$\therefore s(v_{i,k}^{\vee}) = 0.$$

That  $s(\wedge\text{longwires}) = \{1\}$  follows from the duality of  $OR$  and  $AND$ . []

**Lemma 4.3:**

$$\forall X \in \{0,1\}^n, 1 \leq i \leq \log n, 1 \leq k \leq \frac{n}{2^i},$$

$$(a) s(e_{i,k}^{\vee}) = 1 \Rightarrow s(\vee\text{longwires under } e_{i,k}^{\vee}) = \{0\},$$

$$(b) s(e_{i,k}^{\wedge}) = 0 \Rightarrow s(\wedge\text{longwires under } e_{i,k}^{\wedge}) = \{1\}.$$

**Proof:**

$$(a) s(e_{i,k}^{\vee}) = 1 \Rightarrow s(e_{i,k}^{\vee}) = 0 \text{ by Fact 4} \\ = s(v_{i-1,2k-1}^{\vee})$$

$$\text{But } s(v_{i-1,2k-1}^{\vee}) = OR(x_{i-1}, \dots, x_{i-2}) = 0$$

$$\therefore \text{ by Lemma 4.2, } s(\vee\text{longwires under } e_{i,k}^{\vee}) = \{0\}.$$

(b) follows by the duality of  $OR$  and  $AND$ . []

**Definitions:**

A legal state of circuit  $SOR/SAND(n)$  is called a  $k$ -state if  $OR(n) = k$ , for  $k \in \{0, 1\}$ . If  $G = (V, E)$  is a circuit embedded in  $GD_{IJ}$  and  $k \in \{0, 1\}$ , then the  $k$ -area of  $G$  (or  $E$ ) is  $(\lambda \sum_{w \in E \text{ \& } s(w)=k} \|w\|)$ , where  $\lambda$  is a technology dependent constant  $> 0$ .

**Lemma 4.4:**

For any 1-state of circuit  $SOR/SAND(n)$ , the 1-area of  $\Psi(\vee T(1, n)) \leq 10n$  and the 0-area of  $\Psi(\wedge T(1, n)) \leq 10n$ .

**Proof:**

Consider  $\Psi(\vee T(1, n))$ .

Let  $A_{\vee L}(n)$  be the 1-area of  $\vee longwires$  in layout  $LF(n)$ . Recall from Lemma 4.3 that for  $w \in \vee longwire$ ,  $s(w) = 1 \Rightarrow s(\vee longwires \text{ under } w) = \{0\}$ . Hence the following recurrence for  $A_{\vee L}$ .

$$A_{\vee L}(2) \leq 1 + \sqrt{2}$$

$$A_{\vee L}(n) \leq A_{\vee L}\left(\frac{n}{2}\right) + n + \sqrt{2} - 1$$

$$\leq 2n$$

Recall from Lemma 4.1 that the  $\vee shortwires$  can contribute at most  $10n$  1-area. Hence the first part of the Lemma. By duality of  $OR$  and  $AND$ , and by the symmetry of  $LF$ , the 0-area of  $\Psi(\wedge T(1, n)) \leq 10n$ .  $\square$

**Theorem 4.1:**

For all pairs of legal states, the worst case energy used by  $LF(n)$ ,  $E_{worst}^U(LF(n)) = O(n)$ .

**Proof:**

Let  $s_1$  and  $s_2$  be two legal states of  $LF$ . There are 3 cases in which  $LF$  consumes energy when  $LF$  switches from  $s_1$  to  $s_2$ .

1)  $s_1$  is a 0-state and  $s_2$  is a 1-state. By Lemma 4.2,  $s_1$  a 0-state  $\Rightarrow s(\vee longwires) = \{0\}$  and  $s(\wedge longwires) = \{1\}$ . By Lemma 4.4, when  $LF$  switches to state  $s_2$ , at most  $10n$  area of  $\Psi(\vee T(1, n))$  switches on and at most  $10n$  area of  $\Psi(\wedge T(1, n))$  switches off, consuming at most  $20n$  energy.

2)  $s_1$  is a 1-state and  $s_2$  is a 0-state. Theorem 4.1 follows as above except the 1-area of  $\Psi(\vee T(1, n))$  switches off and the 0-area of  $\Psi(\wedge T(1, n))$  switches on.

3)  $s_1$  and  $s_2$  are both 1-states and  $s_1 \neq s_2$ . Clearly, at most twice the switching occurs in this case as above, using at most  $40n$  energy.  $\square$

**4.2 Compare Functions**

The technique developed in the previous section to yield energy-efficient VLSI circuits for  $OR$  and  $AND$  functions can be applied further. In particular, this section shows how to extend the technique to produce energy-efficient circuits that compare boolean bit strings lexicographically.

**Definition:**

Let  $A = (a_1, \dots, a_n)$  and  $B = (b_1, \dots, b_n)$  such that  $a_i, b_i \in \{0, 1\}$  for  $1 \leq i \leq n$ . Then  $A = B$  iff  $\forall i \Rightarrow 1 \leq i \leq n, a_i = b_i$ .

## Upper and Lower Bounds on Switching Energy in VLSI

### Theorem 4.2:

$A = B$  can be computed by a VLSI circuit  $C_n$  of  $O(\log n)$  depth and  $E_{worst}(C_n) = \Theta(n)$  when the  $n$  inputs are on a convex boundary of  $C_n$ .

### Proof:

$$A = B \text{ iff } [\bigwedge_{1 \leq i \leq n} (a_i = b_i)] = 1$$

$$\text{iff } [\bigwedge_{1 \leq i \leq n} \overline{(a_i \oplus b_i)}] = 1$$

Embed  $a_i$  and  $b_i$  such that they are  $O(1)$  distance apart. Clearly  $\overline{(a_i \oplus b_i)}$  can be computed in  $O(1)$  energy. Thus, by Theorem 4.1,  $\bigwedge_{1 \leq i \leq n} \overline{(a_i \oplus b_i)}$  can be computed in  $E_{worst} = O(n)$  when the circuit depth is  $O(\log n)$  and  $a_i, b_i$  are on a convex boundary of the layout.  $\square$

To obtain energy-efficient VLSI circuits for compare functions such as  $>$ ,  $\leq$ , etc., (eg. Is  $X > Y$  where  $X = (x_1, \dots, x_n)$  and  $Y = (y_1, \dots, y_n)$  are lexicographic boolean bit strings), a VLSI circuit  $EG$ , illustrated in Figure 4.3 below, is constructed from three modified instances of the *SOR/SAND* circuit and some connecting circuitry.  $EG$  contains an instance of *SOR/SAND* as described in section 4.1, and a subcircuit called *SAND/SOR* in which the logical gates are reversed. ie.  $\wedge$ -gates in *SOR/SAND* become  $\vee$ -gates in *SAND/SOR* and vice versa. The third "plane" of circuitry resembles "half" a *SOR/SAND* circuit. This partial *SOR/SAND*-like subcircuit, indicated by the striped lines in Figure 4.3, is made energy-efficient by "piggybacking" off the complete *SOR/SAND* subcircuit.

### Theorem 4.3:

The worst case energy used by  $EG(x_n, y_n, \dots, x_1, y_1)$ ,  $E_{worst}^U(EG(n)) = O(n)$ .

**Proof Idea:** Intuitively, each *SOR/SAND*-like "plane" uses  $O(n)$  energy by Theorem 4.1. The connecting wires (denoted by the textured lines in Figure 4.3) use area  $O(n)$  and hence energy  $O(n)$ . The combined circuit thus uses only  $O(n)$  energy in the worst case. The formal details of the construction and analysis can be found in [Ki87].

## 4.3 Recent Results

Kissin et al [KKT90] recently extended the *SOR/SAND* technique to  $k$ -threshold functions. They have obtained a linear upper bound on the worst case uniswitch energy required to compute a  $k$ -threshold function, where  $k$  is a fixed constant.

## 5. AVERAGE ENERGY

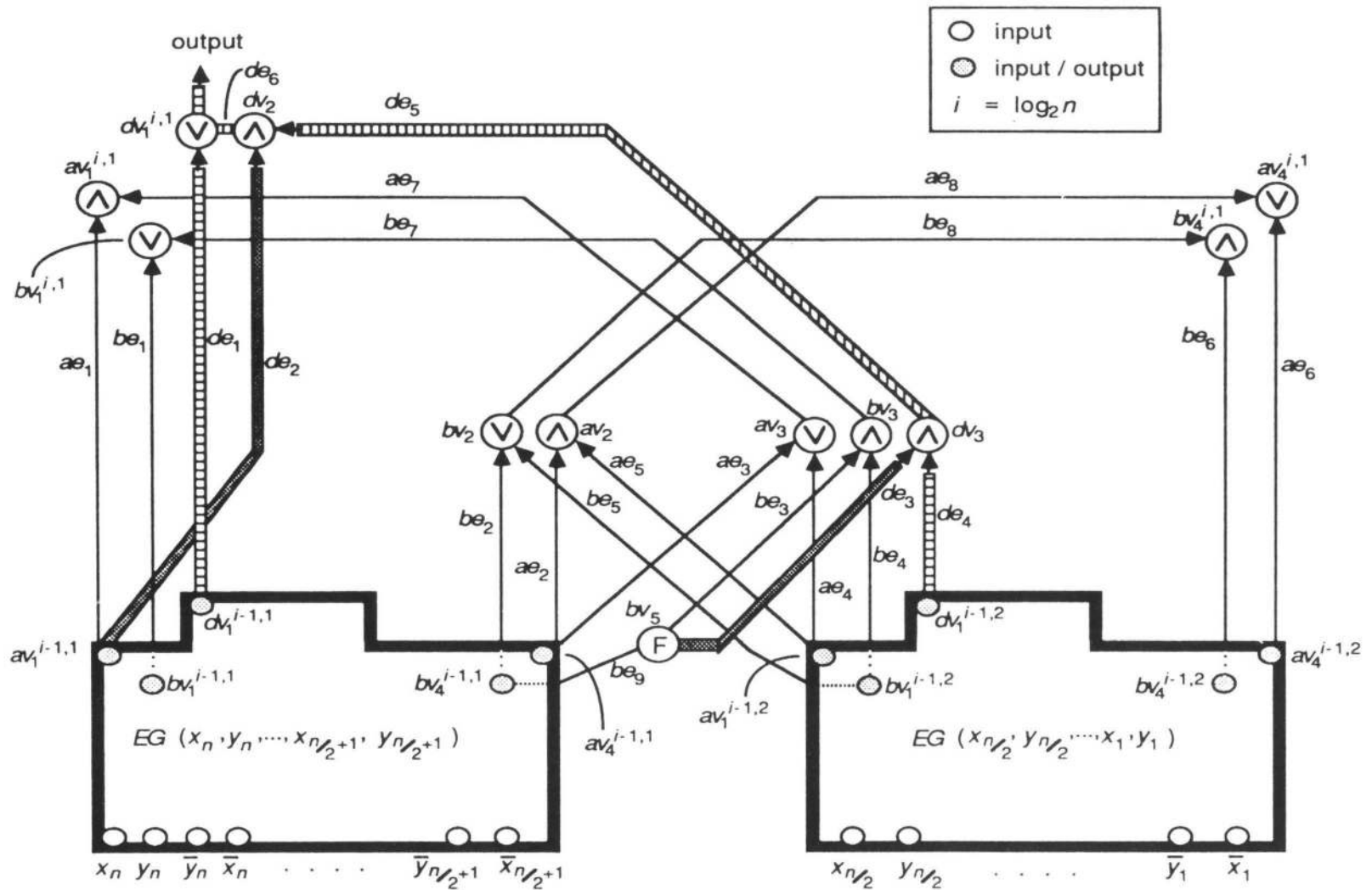
### 5.1 Definitions and Easy Bounds

This section derives the basic definitions needed to discuss average energy, and analyzes some simple circuits. In particular, an  $n$ -leaf complete tree whose interior nodes are either all  $\wedge$ -gates or all  $\vee$ -gates is shown to use  $O(n)$  average energy, when the leaves are embedded on a convex boundary of the layout. A novel adder layout that uses linear average energy is described in section 5.2.

Recall from section 2 the following definition of average switching energy.

### Definition:

If  $E_w(C_n, s_0, X)$  (ref. section 2) is the wire energy dissipated when  $C_n : s_0 \rightarrow X$ , then  $E_a(C_n)$ , the average case switching energy is given by



Most  $i, k$  superscripts are omitted for clarity.

Figure 4.3.  $EG(x_n, y_n, \dots, x_1, y_1)$

## Upper and Lower Bounds on Switching Energy in VLSI

$$E_a(C_n) \triangleq \sum_{(s_0, X)} E_w(C_n, s_0, X) / 2^{2n}$$

where  $n = |X|$  and  $2^{2n}$  is the number of input pairs.  $E_a$  is also written as a function of  $n$ , ie.  $E_a(n)$ .

$E_a(C_n)$  averages the wire energy over all pairs of inputs, which are assumed to be equally likely. Note that the definition of  $E_a$  in [Ki82] averages the wire energy over all inputs to a circuit in a particular state. Thus, the definition above is stronger in that a lower bound for  $E_a(C_n)$  does not depend on picking a "bad" initial state.

Average switching is defined analogously to average energy as follows.

### Definition:

If  $Sw(C_n, s_0, X)$  is the number of wires in circuit  $C_n$  that switch when  $C_n : s_0 \rightarrow X$ , then  $Sw_a(C_n)$ , the *average switching* is given by

$$Sw_a(C_n) = \sum_{(s_0, X)} Sw(C_n, s_0, X) / 2^{2n}$$

where  $n = |X|$  and  $2^{2n}$  is the number of input pairs.  $Sw_a$  is also written as a function of  $n$ , ie.  $Sw_a(n)$ . Note that  $Sw_a(C_n)$  averages the *number* of wires that switch, while  $E_a(C_n)$  averages the *area* of the wires of  $C_n$  that switch. Implicit in the definitions of  $E_a(C_n)$  and  $Sw_a(C_n)$  is the assumption that the inputs to  $C_n$  are uniformly distributed over  $\{0,1\}$ .

### Definitions:

Let  $w = (v, v')$  be a directed edge (wire) of circuit  $C$ . If  $L$  is the length of the longest path from any input to node  $v$ , then edge  $w$  is at *level*  $L$ . For consistency, an input node is called a *level 0* wire. A complete binary tree with  $n$  leaves and an  $\vee$ -gate ( $\wedge$ -gate) at each node is called an  $n$ -OR ( $n$ -AND) *tree*.

The following analysis shows that the average energy used by an  $n$ -AND tree or an  $n$ -OR tree is  $O(n)$  when the inputs are uniformly distributed over  $\{0,1\}$ .

### Switching Lemma 5.1:

An  $n$ -OR ( $n$ -AND) tree  $T_n$  has average switching  $Sw_a(n) = \Theta(n)$ .

### Proof:

$Sw_a(n) = O(n)$  follows from the fact that  $T_n$  contains  $O(n)$  wires.

$Sw_a(n) = \Omega(n)$  follows from the fact that  $Pr(\text{level 0 wire switches}) = 1/2$ . Thus,  $n/2$  level 0 wires switch on the average.  $\therefore Sw_a(n) = \Theta(n)$ .  $\square$

### Theorem 5.1:

There exists a layout of an  $n$ -OR ( $n$ -AND) tree  $T_n$  with leaves on a convex boundary, which consumes average energy  $E_a(n) = \Theta(n)$ .

### Proof:

By [BK82] a complete tree embedded with  $n$  leaves on a convex boundary requires  $\Omega(n \log n)$  area. Consider  $\hat{T}_n$ , a standard embedding illustrated in Figure 5.1. Input node  $i$  is at position  $(i, 0)$ , so that all  $\vee$ -gates have an x-coordinate  $k + \frac{1}{2}$ , for  $i, k \in \mathbb{N}$ . Since the vertical wire segments of  $\hat{T}_n$  contribute only  $O(n)$  area, they are omitted from the following analysis. Let  $A(k)$  denote the area of the horizontal wires of  $\hat{T}_k$ .  $A(k)$  is determined by the following recurrence.  $n$  is a power of 2.

$$A(2) = 1$$

$$A(n) = 2A\left(\frac{n}{2}\right) + \frac{n}{2}$$

## Upper and Lower Bounds on Switching Energy in VLSI

The probability,  $Pr$ , that a wire of  $T_n$  switches is as follows:

$Pr$  ( level 0 wire (ie. input) switches) =  $1/2$

$Pr$  ( level  $k$  wire  $w$  switches) =  $Pr ( w : 0 \rightarrow 1 \text{ or } w : 1 \rightarrow 0 ) = \frac{2^{2^k} - 1}{2^{2^{k+1} - 1}}$

Let  $E_a(k)$  denote the average energy of (the horizontal wires of)  $\hat{T}_k$ .  $E_a(k)$  is obtained recursively below from the switching probabilities and the area recurrence for  $A$ .

$E_a(2) = 1/2$

$E_a(n) = 2E_a(\frac{n}{2}) + \frac{n}{2}(\frac{2^{n/2} - 1}{2^{n-1}})$

where  $\frac{n}{2}(\frac{2^{n/2} - 1}{2^{n-1}})$  is the horizontal wire area at level  $(\log n) - 1$  times the probability that this area switches.

$n \rightarrow \infty \Rightarrow \frac{n}{2}(\frac{2^{n/2} - 1}{2^{n-1}}) \rightarrow 0. \therefore E_a(n) = O(n)$ .

$E_a(n) = \Omega(n)$  follows from the fact that  $Pr$  ( level 0 wire switches) =  $1/2$ . Thus, the average number of level 0 wires that switch is  $n/2$ . Since each input wire is at least 1 unit long,  $E_a(n) = \Omega(n)$ .  $\therefore E_a(n) = \Theta(n)$ .  $\square$

### 5.2 An Average Energy-Efficient Adder Layout

The Brent/Kung layout [BK82] of a shallow depth parallel prefix adder [LF80] uses  $O(n \log n)$  energy both in the worst case and average case. While section 3 showed that this is optimal in the worst case, this section shows that, on the average, one can do better.

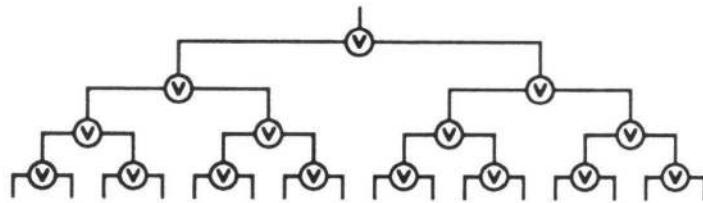
The recursive construction described in this section uses a layout technique of section 4 to obtain an embedding of the parallel prefix adder that uses, on the average,  $O(n)$  energy. The following definitions introduce terminology that is later used to describe layouts.

#### Definitions:

Let  $p$  be a node in a VLSI circuit  $A$ , embedded at location  $(x_p, y_p)$ , and let  $q$  be a node in  $A$  embedded at coordinate  $(x_q, y_q)$ . Let  $d \in \mathbf{Z}^+$ .

(i)  $q$  is @  $p$  if  $(x_q, y_q) = (x_p, y_p)$ .

(ii)  $q$  is  $d$  units north (also written as  $N$ ) of  $p$  if  $(x_q, y_q) = (x_p, y_p + d)$ .



An  $O(n \log n)$  layout for an  $n$ -OR tree

Figure 5.1.

## Upper and Lower Bounds on Switching Energy in VLSI

(iii)  $q$  is  $d$  units south ( $S$ ) of  $p$  if  $(x_q, y_q) = (x_p, y_p - d)$ .

(iv)  $q$  is  $d$  units east ( $E$ ) of  $p$  if  $(x_q, y_q) = (x_p + d, y_p)$ .

(v)  $q$  is  $d$  units west ( $W$ ) of  $p$  if  $(x_q, y_q) = (x_p - d, y_p)$ .

(vi)  $q$  is  $(d\sqrt{2})$  units north-east ( $NE$ ) of  $p$  if  $(x_q, y_q) = (x_p + d, y_p + d)$ .

(vii)  $q$  is  $(d\sqrt{2})$  units south-east ( $SE$ ) of  $p$  if  $(x_q, y_q) = (x_p + d, y_p - d)$ .

$d$  is called the *displacement*. An element of  $\{N, S, E, W, NE, SE, @\}$  is called a *heading*. Headings are also defined on coordinates directly. For example,  $q$  is two units  $S$  of  $(x_1, y_1)$  if  $(x_q, y_q) = (x_1, y_1 - 2)$ . Clearly  $d=0$  for the heading  $@$ .

Let  $S(n)$  denote a VLSI circuit (ie. an embedded circuit) that computes the carries generated by adding two  $n$ -bit binary numbers. In particular,  $S(n)$  receives as input two vectors,  $(p_1, \dots, p_n)$  and  $(g_1, \dots, g_n)$  where  $p_i = a_i \oplus b_i$  and  $g_i = a_i \wedge b_i$  for  $1 \leq i \leq n$ , and  $\sum_{i=1}^n a_i 2^{i-1}$  and  $\sum_{i=1}^n b_i 2^{i-1}$  are the two  $n$ -bit binary numbers being added.  $S(n)$  produces as output the carry vector  $(c_1, \dots, c_n)$  defined recursively as  $c_0 = 0$ ,  $c_i = g_i \vee (p_i \wedge c_{i-1})$  for  $1 \leq i \leq n$ . Once the carries are computed, the sum vector  $(s_1, \dots, s_{n+1})$ , defined as  $s_i = p_i \oplus c_{i-1}$  for  $1 \leq i \leq n$ , and  $s_{n+1} = c_n$ , can be computed.

$S(n)$  is defined recursively. An abstract view of  $S(n)$  called *generic*  $S(n)$  is shown in Figure 5.5. Four corners of  $S(n)$  are distinguished, moving clockwise from the bottom left corner, as  $BLC$ ,  $TLC$ ,  $TRC$ , and  $BRC$ , all of which denote the coordinates of the gate or I/O port embedded at the respective corner. Computed at these corners of  $S(n)$  are the following: carry generate  $g_{i+n-1}$  at  $BLC$ ; carry propagate  $p_i$  at  $BRC$ ; block generate  $G(n, i)$ , which is the carry generated by the inputs to  $S(n)$ , at  $TLC$ ; and block propagate  $P(n, i) = (p_i \wedge p_{i+1} \wedge \dots \wedge p_{i+n-1})$

$S(n) = (LV(n), LE(n))$  is a VLSI circuit composed of  $LV(n)$ , the set of embedded nodes and  $LE(n)$ , the set of embedded edges. Each element of  $LV(n)$  is written as a triple  $(v, f, L)$ , where  $v$  is

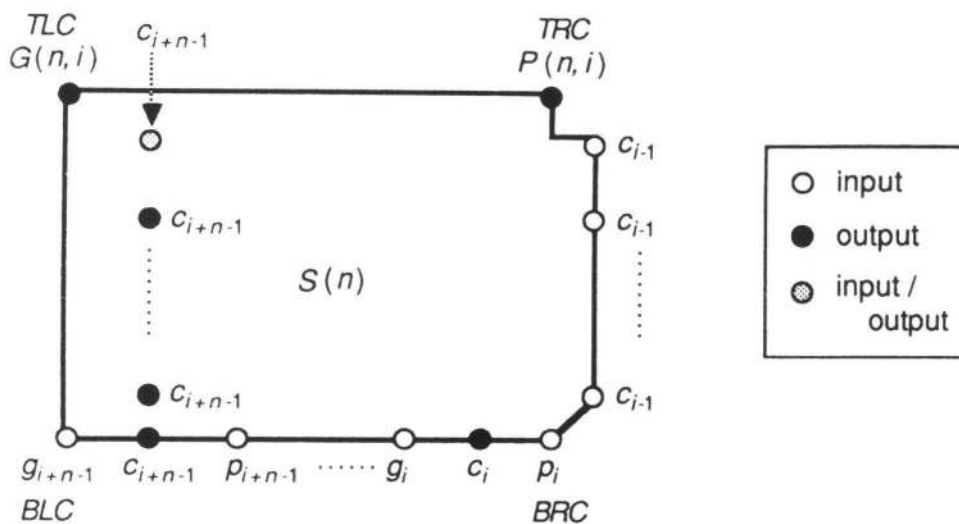


Figure 5.5. *generic*  $S(n)$

## Upper and Lower Bounds on Switching Energy in VLSI

the node identifier,  $f$  is the function computed at node  $v$ , and  $L$  is the location of  $v$  in the embedding.  $L$  is a triple  $(k, D, v_p)$ , where  $k \in \mathbb{Z}^+$  is a displacement,  $D \in \{N, S, E, W, NE, SE, @\}$  is a heading, and  $v_p$  is a coordinate of the layout or a node whose location is already defined. The location of  $v$  is determined relative to  $v_p$ . For input nodes, the function entry in the node triple is  $I$ . In  $S(n)$ , each node labeled  $c_i$  computes the identity function. These nodes are output ports that are functionally denoted as  $O$  in the node triple of  $LV(n)$ . Examples of elements of  $LV(n)$  are:

- (i)  $(g_2, I, (0, @, BLC(S(2))))$  states that  $g_2$  is an input node embedded at the bottom left corner of  $S(2)$ .
- (ii)  $(v_1, \vee, (2, N, g_2))$  means that  $v_1$  is an  $\vee$ -gate embedded 2 units north of input  $g_2$ .
- (iii)  $(c_1, O, (1, W, BRC(S(2))))$  states that  $c_1$  is an output port embedded 1 unit west of the bottom right corner of  $S(2)$ .

An element of  $LE(n)$  is written as a pair  $(w, k)$  where  $w$  is an edge, written as a pair of adjacent nodes, and  $k \in \mathbb{Z}^+$  is the length of  $w$  in the embedding.

Recall that  $S(n)$  is a VLSI circuit that computes the carries of binary addition. This section has thus far described the tools used in the following recursive definition of  $S(n)$ .

### Recursive Definition of $S(n)$ :

**base case:**  $n = 2$ , illustrated in Figure 5.6.

$S(2) = (LV(2), LE(2))$  where the node set  $LV(2)$  and the edge set  $LE(2)$  are as follows.

$$\begin{aligned}
 LV(2) = \{ & \text{for } i \in \{1, 3, 5, \dots, n-1\}, (p_i, I, (0, @, BRC(S(2))))), \\
 & (g_i, I, (2, W, p_i)), (p_{i+1}, I, (4, W, p_i)), (g_{i+1}, I, (6, W, p_i)), \\
 & (c_i, O, (1, W, p_i)), (v_1, \vee, (2, N, g_{i+1})), \\
 & (v_2, \wedge, (1, N, g_i)), (v_3, \wedge, (2, N, p_i)), (v_4, \vee, (1, N, c_i)), \\
 & (v_5, \wedge, (2, E, v_4)), (c_{i-1}, O, (2, E, v_5)) \} \\
 LE(2) = \{ & \text{for } i \in \{1, 3, 5, \dots, n-1\}, ((g_{i+1}, v_1), 2), ((p_i, v_3), 2), ((v_5, v_4), 2), \\
 & ((p_{i+1}, v_2), 1 + \sqrt{2}), ((g_i, v_2), 1), ((v_4, c_i), 1), ((c_{i-1}, v_5), 2), \\
 & ((g_i, v_4), \sqrt{2}), ((p_i, v_5), \sqrt{2}), ((v_2, v_1), 3 + \sqrt{2}), ((p_{i+1}, v_3), 4 + \sqrt{2}) \}
 \end{aligned}$$

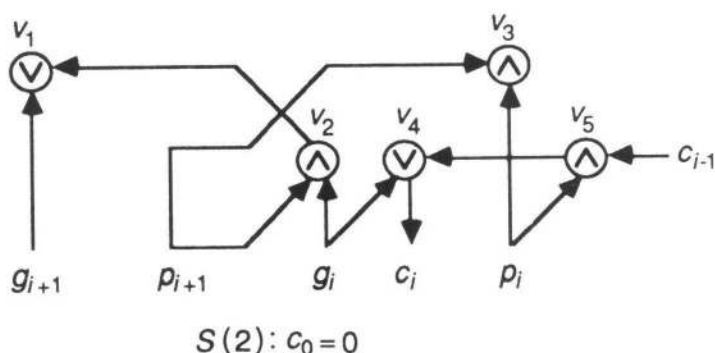


Figure 5.6. Base Case for Adder Circuit



## Upper and Lower Bounds on Switching Energy in VLSI

$S(2)$  is illustrated in Figure 5.6.

**induction step:** illustrated in Figure 5.7.

$S(n)$  is defined recursively from two instances of  $S(\frac{n}{2})$ , distinguished as  $LS(\frac{n}{2})$  for the leftmost instance, and  $RS(\frac{n}{2})$  for the rightmost instance. The labels  $LS$  and  $RS$  are used only where ambiguity might arise; otherwise,  $S$  is used. Input node  $p_i$  is embedded at  $BRC(RS(\frac{n}{2}))$ , and  $g_{i+(n/2)-1}$  is at  $BLC(RS(\frac{n}{2}))$ . Embedded at  $BRC(LS(\frac{n}{2}))$  is input  $p_{i+(n/2)}$ , and at  $BLC(LS(\frac{n}{2}))$  is  $g_{i+n-1}$ .  $LS(\frac{n}{2})$  and  $RS(\frac{n}{2})$  are laid out so that  $BRC(LS(\frac{n}{2}))$  is two units west of  $BLC(RS(\frac{n}{2}))$ .

Because many nodes in  $S(n)$  have the same name, the following auxiliary names are introduced. Let  $\{w_1, w_2, w_3, w_4\}$  denote the nodes at the top four corners of  $LS(\frac{n}{2})$  and  $RS(\frac{n}{2})$ . In particular,  $w_1$  is at  $TLC(LS(\frac{n}{2}))$ ,  $w_2$  is at  $TRC(LS(\frac{n}{2}))$ ,  $w_3$  is at  $TLC(RS(\frac{n}{2}))$ , and  $w_4$  is at  $TRC(RS(\frac{n}{2}))$ . Let  $w_5$  denote the node located  $\sqrt{2}$  units south-east of  $w_2$ .

Recall that  $S(n) = (LV(n), LE(n))$  where  $LV(n)$ , the vertex set, and  $LE(n)$ , the edge set, are defined recursively as follows.

$LV(n) = (LLV(\frac{n}{2}) \cup RLV(\frac{n}{2}) \cup LA(n))$  where  $LLV(\frac{n}{2})$  is the vertex set of  $LS(\frac{n}{2})$ ,  $RLV(\frac{n}{2})$  is the vertex set of  $RS(\frac{n}{2})$ , and

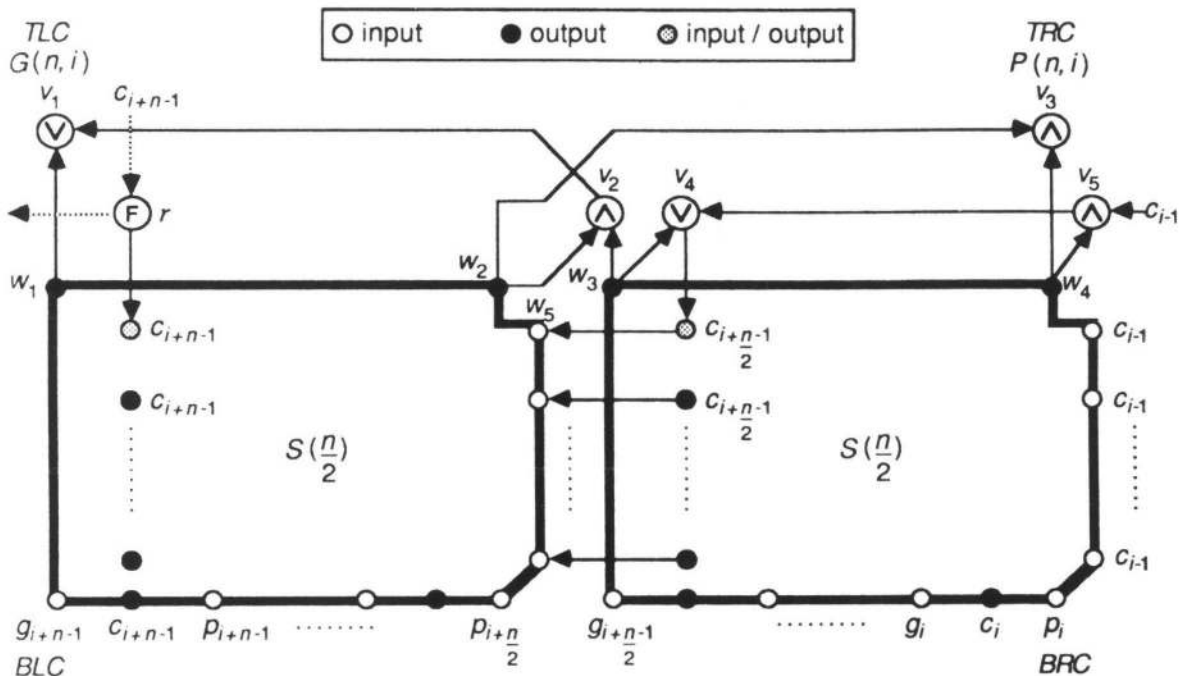


Figure 5.7. Adder Layout  $S(n)$

## Upper and Lower Bounds on Switching Energy in VLSI

$$LA(n) = \{(v_1, \vee, (2, N, w_1)), (v_2, \wedge, (1, N, w_3)), (v_3, \wedge, (2, N, w_4)), \\ (v_4, \vee, (\sqrt{2}, NE, w_3)), (v_5, \wedge, (\sqrt{2}, NE, w_4)), (v_6, F, (\sqrt{2}, NE, w_1))\}$$

$v_6$  is called the *root*( $S(n)$ ) and alternately labeled  $r$ .  $r_L$  denotes *root*( $LS(\frac{n}{2})$ ) and  $r_R$  denotes *root*( $RS(\frac{n}{2})$ ).

$LE(n) = (LLE(\frac{n}{2}) \cup RLE(\frac{n}{2}) \cup LB(n))$  where  $LLE(\frac{n}{2})$  is the edge set of  $LS(\frac{n}{2})$ ,  $RLE(\frac{n}{2})$  is the edge set of  $RS(\frac{n}{2})$ .  $LB(n)$ , the subset of edges that connect subcircuits  $LS(\frac{n}{2})$  and  $RS(\frac{n}{2})$ , is defined as follows.

$$LB(n) = \{((w_1, v_1), 2), ((w_4, v_3), 2), ((w_2, v_2), \sqrt{2} + 1), ((w_3, v_2), 1), \\ ((w_3, v_4), \sqrt{2}), ((v_4, r_R), 2), ((w_4, v_5), \sqrt{2}), ((r_R, w_5), 2), \\ ((r, r_R), 2), ((v_2, v_1), 2n + \sqrt{2} - 1), ((w_2, v_3), 2n + \sqrt{2}), ((v_5, v_4), 2n - 2)\}$$

$S(n)$  is illustrated in Figure 5.7.

In the rest of this section,  $S(n)$  is shown to use  $O(n)$  average energy. Intuitively, the proof begins by observing that a subcircuit of  $S(n)$  is an  $n-AND$  tree. By a proof similar to that of Theorem 5.1, this  $n-AND$  tree subcircuit uses  $O(n)$  average energy. The rest of the proof proceeds very much like the analysis of the *SOR/SAND* circuit of section 4. The wires of  $S(n)$  minus the  $n-AND$  tree subcircuit are partitioned into *shortwires* and *longwires*. The *shortwires* are shown to occupy  $O(n)$  area and are thus eliminated from further discussion. The switching probabilities of the *longwires* are then determined, assuming the inputs are uniformly distributed over  $\{0,1\}$ . Using these switching probabilities, the average energy of the *longwires* is shown to be  $O(n)$ .

### Lemma 5.2:

The VLSI circuit  $S(n)$ , which computes  $n+1$  addition carries in  $O(\log n)$  depth, uses average energy  $E_a(n) = O(n)$ .

### Proof:

The first part of the proof extracts a subcircuit,  $A$ , of  $S(n)$  that is an  $n-AND$  tree. Let  $V_3$  be the set of all nodes of  $S(n)$  labeled  $v_3$ . Let  $W_{23}$  be the set of edges of  $S(n)$  labeled  $(w_2, v_3)$  and let  $W_{43}$  be the set of edges of  $S(n)$  labeled  $(w_4, v_3)$ . Let  $P = (p_1, \dots, p_n)$  be inputs of  $S(n)$ . Let  $A = (AN, AE)$  denote a subcircuit of  $S(n)$  such that  $AN = V_3 \cup P$  and  $AE = W_{23} \cup W_{43}$ . It is easy to see that  $A$  is an  $n-AND$  tree. Since the inputs  $P$  are uniformly distributed over  $\{0,1\}$ ,  $A$  uses average energy  $O(n)$  by a proof similar to that of Theorem 5.1.

The analysis that follows examines  $\hat{S}(n)$ , which is  $S(n)$  minus the  $n-AND$  tree  $A$ . Let  $\hat{S}(n) = (L\hat{V}, L\hat{E})$  denote the graph obtained by removing the nodes and edges of  $A$  from  $S(n)$ . In particular,  $L\hat{V} = LV(n) - AN$  and  $L\hat{E} = LE(n) - AE$ .

The reader is advised to refer to Figure 5.7 to clarify the following definitions of *longwires* and *shortwires*.

### Definitions:

Let *blongwires*( $\hat{S}(n)$ ) be the set of wires of  $\hat{S}(n)$  labeled  $(v_2, v_1)$ .

Let *rlongwires*( $\hat{S}(n)$ ) be the set of wires of  $\hat{S}(n)$  labeled  $(v_5, v_4)$ .

*longwires*( $\hat{S}(n)$ )  $\triangleq$  *blongwires*( $\hat{S}(n)$ )  $\cup$  *rlongwires*( $\hat{S}(n)$ )

*shortwires*( $\hat{S}(n)$ )  $\triangleq$   $L\hat{E} - \text{longwires}(\hat{S}(n))$

An element of *longwires*( $\hat{S}(n)$ ) is called a *longwire*. Similarly, *blongwire* is a member of *blongwires*( $\hat{S}(n)$ ) and *rlongwire* is a member of *rlongwires*( $\hat{S}(n)$ ).

## Upper and Lower Bounds on Switching Energy in VLSI

Let  $R(n)$  be the area of *shortwires* ( $\hat{S}(n)$ ).

Let  $L(n)$  be the area of *longwires* ( $\hat{S}(n)$ ).

The following Lemma shows that the area of *shortwires* ( $\hat{S}(n)$ ) is  $O(n)$ .

**Lemma 5.3:**

$$R(n) = O(n)$$

**Proof:**

$$R(2) = 9 + 3\sqrt{2}$$

$$R(n) \leq 2R\left(\frac{n}{2}\right) + 8\log n + 8 + 3\sqrt{2}$$

$$= O(n)$$

The  $8\log n$  factor in the expression above accounts for the wire area needed to fanout intermediate carries  $c_{i-1}$  and  $c_{i+(n/2)-1}$  (see Figure 5.7). []

Since the *shortwires* can contribute at most  $O(n)$  energy, the remaining analysis considers only the *longwires*. The area of the *longwires* can be determined from the following recurrence.

$$L(2) = 5 + \sqrt{2}$$

$$L(n) = 2L\left(\frac{n}{2}\right) + 4n + \sqrt{2} - 3$$

The reader can verify that  $L(n) = O(n \log n)$ .

It remains to show that the average energy of the *longwires* is  $O(n)$ , which follows from examining the switching probabilities of the *longwires*. Note from Figure 5.7 that the head of a *longwire* is a conjunction labeled either  $v_5$  or  $v_2$ , and in both cases, a node of the  $n$ -AND tree is one of the conjuncts. Thus, it is intuitively clear that the *longwires* have a small probability of switching since the *longwires* are even less likely to switch than the wires of the  $n$ -AND tree. The following discussion formalizes this intuition.

Recall from sections 2 and 4.1 that, for  $z \in L\hat{V} \cup L\hat{E}$ ,  $s(z)$  denotes the value of  $z$  for some input to  $\hat{S}(n)$ . The label  $z$ , however, serves double duty in the following analysis.  $z$  denotes a node (or wire) and its respective function. The intended usage is clear from the context. The following definition of *stage* facilitates the discussion of the switching probabilities of wires in  $\hat{S}(n)$ .

**Definitions:**

A *blongwire* of area  $2n + \sqrt{2} - 1$  is a *stage*  $(\log n) - 1$  *blongwire*. An *rlongwire* of area  $2n - 2$  is a *stage*  $(\log n) - 1$  *rlongwire*. A node  $x$  is at *stage*  $k$ , denoted  $x^k$ , if  $x$  is the tail of a *stage*  $k$  *longwire* or  $x$  is an input to the tail of a *stage*  $k$  *longwire*.

Let  $Pr(w_b^k \text{ switches})$  denote the probability that *stage*  $k$  *blongwire*  $w$  switches. Let  $Pr(w_r^k \text{ switches})$  denote the probability that *stage*  $k$  *rlongwire*  $w$  switches.

The following Lemma shows that the probability of a *longwire* switching decreases as the *stage* of the wire increases.

**Lemma 5.4:**

$$(i) Pr(w_b^k \text{ switches}) \leq \frac{1}{2^{2^k - 1}}$$

$$(ii) Pr(w_r^k \text{ switches}) \leq \frac{1}{2^{2^k - 1}}$$

**Proof:**

$$(i) Pr(w_b^k \text{ switches}) = Pr(w_b^k: 0 \rightarrow 1) + Pr(w_b^k: 1 \rightarrow 0) \\ = 2Pr(w_b^k: 1 \rightarrow 0) \leq 2Pr(s(w_b^k) = 1)$$

$$\leq 2 Pr(s(v_3^k) = 1) * Pr(s(v_1^k) = 1)$$

Since  $v_3^k = \bigwedge_{i=j}^{j+2^k-1} p_i$  for some  $j \Rightarrow 1 \leq j \leq n-2^k+1$ ,

$$Pr(s(v_3^k) = 1) = Pr(\bigwedge_{i=j}^{j+2^k-1} p_i = 1) = \frac{1}{2^{2^k}}$$

$$\therefore Pr(w_b^k \text{ switches}) \leq 2\left(\frac{1}{2^{2^k}}\right) * Pr(s(v_1^k) = 1) \leq \frac{1}{2^{2^k-1}}$$

(ii) By the same argument as (i) above,

$$Pr(w_r^k \text{ switches}) \leq 2 Pr(s(v_3^k) = 1) * Pr(s(v_6^k) = 1)$$

$$\text{From (i) above, } Pr(s(v_3^k) = 1) = \frac{1}{2^{2^k}}$$

$$\therefore Pr(w_r^k \text{ switches}) \leq 2\left(\frac{1}{2^{2^k}}\right) * Pr(s(v_6^k) = 1) \leq \frac{1}{2^{2^k-1}} \quad \square$$

Continuation of Proof of Lemma 5.2 (ie.  $E_a(S(n)) = O(n)$ ).

Let  $SB(n)$  denote the average switching of *blongwires* ( $\hat{S}(n)$ ).

Let  $SR(n)$  denote the average switching of *rlongwires* ( $\hat{S}(n)$ ).

Let  $EB(n)$  denote the average energy of *blongwires* ( $\hat{S}(n)$ ).

Let  $ER(n)$  denote the average energy of *rlongwires* ( $\hat{S}(n)$ ).

The following Lemma shows that the average switching of the *longwires* ( $\hat{S}(n)$ ) is  $O(n)$ . However, more important are the recurrences used to obtain the average switching. These recurrences are subsequently used in Lemma 5.6 to show that the average energy of the *longwires* ( $\hat{S}(n)$ ) is  $O(n)$ .

**Lemma 5.5:**

(i)  $SB(n) = O(n)$

(ii)  $SR(n) = O(n)$

**Proof:**

(i)  $SB(n)$ , the average switching of the *blongwires* ( $\hat{S}(n)$ ), is given by the following recurrence, which uses the probabilities of Lemma 5.4.

$$SB(2) \leq 1$$

$$SB(n) \leq 2SB\left(\frac{n}{2}\right) + \frac{1}{2^{(n/2)-1}}$$

This standard recurrence has the solution  $SB(n) = O(n)$ .

(ii) Although different stage  $k$  *rlongwires* have different switching probabilities, Lemma 5.4 provides an upper bound on these probabilities.

$$SR(2) \leq 1$$

$$SR(n) \leq 2SR\left(\frac{n}{2}\right) + \frac{1}{2^{(n/2)-1}}$$

$$\therefore SR(n) = O(n). \quad \square$$

The following Lemma shows that the average energy of *longwires* ( $\hat{S}(n)$ ) is  $O(n)$ .

**Lemma 5.6:**

(i)  $EB(n) = O(n)$

(ii)  $ER(n) = O(n)$

**Proof:**

(i) Using the recurrences for area and average switching obtained above, the following recurrence is obtained for  $EB(n)$ , the average energy of the *blongwires* ( $\hat{S}(n)$ ).

$$EB(2) \leq 3 + \sqrt{2}$$

$$EB(n) \leq 2EB\left(\frac{n}{2}\right) + \frac{(2n + \sqrt{2} - 1)}{2^{(n/2)-1}}$$

## Upper and Lower Bounds on Switching Energy in VLSI

$$= O(n)$$

(ii)  $ER(n)$ , the average energy of the *rlongwires* ( $\hat{S}(n)$ ), is similarly obtained from the area recurrence and the switching probabilities cited above.

$$ER(2) \leq 2$$

$$ER(n) \leq 2ER\left(\frac{n}{2}\right) + \frac{(2n-2)}{2^{(n/2)-1}}$$

$$= O(n) \quad \square$$

From Lemma 5.3, the *shortwires* ( $S(n)$ ) use  $E_a(n) = O(n)$ . By a proof similar to that of Theorem 5.1, and by Lemma 5.6, the *longwires* ( $S(n)$ ) use  $E_a(n) = O(n)$ . Hence,  $S(n)$ , which computes the carries for adding two  $n$ -bit binary numbers, uses average energy  $E_a(S(n)) = O(n)$ .

[] end of Lemma 5.2.

### Theorem 5.2:

The average energy needed to add two  $n$ -bit boolean numbers in depth  $O(\log n)$  is  $E_a(n) = O(n)$ .

#### Proof:

The VLSI circuit  $S(n)$  can be used to compute the carries  $(c_1, \dots, c_n)$ . Notice that in the layout of  $S(n)$ , for  $1 < i \leq n$ , input  $p_i$  is embedded 3 units away from carry  $c_{i-1}$ . (Assume a constant 0 is located near  $p_1$ .) Hence, each element of the sum vector  $(s_1, \dots, s_{n+1})$  can be computed from the carry vector in  $O(1)$  energy, since  $s_i = p_i \oplus c_{i-1}$  for  $1 \leq i \leq n$ , and  $s_{n+1} = c_n$ . []

## 6. OPEN PROBLEMS:

### *Lower Bounds on Single-Valued Functions*

To date, no nontrivial energy lower bounds are known for single-valued functions. In section 3, a superlinear energy lower bound is derived for the parity function, in the special case where the circuit contains only  $\oplus$ -gates and negations. For an arbitrary basis, the parity problem is open, as is the majority function. The only other results known for single-valued functions are the upper bounds described in section 4.

### *Energy-Efficient Design Techniques*

The design techniques of section 4 were applied to specific functions – *OR*, compare and addition. Characterize a class of functions for which the circuit and layout techniques of section 4 produce energy-efficient designs. Moreover, the layout technique alone, as applied to the parallel prefix adder circuit in section 5, is likely applicable to a larger class of circuits. In both the *SOR/SAND* circuit and the adder circuit, the layout technique entails embedding tree subcircuits in a specific manner.

## ACKNOWLEDGEMENTS:

I am very thankful to Stephen Cook for supervising this research. Cook also proved Lemma 3.2. Les Valiant provided an idea that lead to Lemma 3.2. I thank Charles Rackoff and Joachim von zur Gathen for their contributions to the presentation of this work. The importance of switching energy to VLSI was originally suggested to me by Carver Mead.

### REFERENCES:

[ACR88]

Aggarwal, A., A. Chandra, P. Raghavan, "Energy Consumption in VLSI Circuits", *Proceedings of 20th ACM STOC*, May 1988, pp. 205-216.

[Bo77]

Borodin, A., "On Relating Time and Space to Size and Depth", *SIAM Journal of Computing*, Vol. 6, No. 4, December 1977, pp. 733-744.

[BK82]

Brent, R.P., H.T. Kung, "A Regular Layout for Parallel Adders", *IEEE Transactions on Computers*, Vol. C-31, No. 3, March 1982, pp. 260-264.

[BK81]

Brent, R.P., H.T. Kung, "The Area-Time Complexity of Binary Multiplication", *JACM*, Vol. 28, No. 3, July 1981, pp. 521-534.

[BK80]

Brent, R.P., H.T. Kung, "On the Area of Binary Tree Layouts," *Information Processing Letters*, Vol. 11, No. 1, August 1980, pp. 46-48.

[Ki90]

Kissin, G., "Models of Multiswitch Energy", *CWI Quarterly*, Vol. 3, No. 1, March 1990, pp. 45-66.

[Ki87]

Kissin, G., "Modeling Energy Consumption in VLSI Circuits", *PhD Thesis*, Department of Computer Science, University of Toronto, 1987.

[Ki85]

Kissin, G., "Functional Bounds on Switching Energy", *Proceedings of 1985 Chapel Hill Conference on Very Large Scale Integration*, May 1985, pp. 181-196.

[Ki82]

Kissin, G., "Measuring Energy Consumption in VLSI Circuits: a Foundation", *Proceedings of 14th ACM STOC*, May 1982, pp. 99-104.

[KKTV90]

Kissin, G., E. Kranakis, J. Tromp, P. Vitanyi, "The Energy Complexity of Threshold and Other Functions", *CWI Technical Report*, to appear.

[KR65a]

Krohn, K., J. Rhodes, "Algebraic Theory of Machines. I. Prime Decomposition Theorem for Finite Semigroups and Machines", *Transactions of American Mathematical Society*, Vol. 116, 1965, pp. 450-464.

[KR65b]

Krohn, K., J. Rhodes, "Results on Finite Semigroups", *Proceedings of the National Academy of Science*, USA, Vol. 53, 1965, pp. 499-501.

[Le84]

Leo, J., "Energy Complexity in VLSI", *M.S. Thesis*, University of Nymegen, The Netherlands, February 1984.

## Upper and Lower Bounds on Switching Energy in VLSI

[LF80]

Ladner, R.E., M.J. Fischer, "Parallel Prefix Computation", *JACM*, Vol. 27, No. 4, October 1980, pp. 831-838.

[LM81]

Lengauer, T., K. Mehlhorn, "On the Complexity of VLSI Computations", *Proceedings of CMU Conference on VLSI*, Computer Science Press, October 1981, pp. 89-99.

[MC80]

Mead, C., L. Conway, *Introduction to VLSI Systems*, Addison-Wesley, 1980.

[Me86]

Mead, C., private communication

[Me80]

Mead, C., private communication

[Sa76]

Savage, J.E., *The Complexity of Computing*, John Wiley & Sons, 1976.

[ST86]

Snyder, L., A. Tyagi, "The Energy Complexity of Transitive Functions", *Proceedings of 24th Allerton Conference on Communication, Control and Computing*, October 1986, pp. 562-572.

[Th80]

Thompson, C., "A Complexity Theory for VLSI", *PhD Thesis*, Dept. of Computer Science, Carnegie-Mellon University, 1980.

[Va84]

Valiant, L., private communication

[Vi83]

Vuillemin, J., "A Combinatorial Limit to the Computing Power of VLSI Circuits", *IEEE Transactions on Computers*, Vol. C-30, No. 2, 1983, pp. 135-140.

[Ya81]

Yao, A., "The Entropic Limitations on VLSI Computations", *Proceedings of 13th ACM STOC*, May 1981, pp. 308-311.

