Correspondence

Variants of an Improved Carry Look-Ahead Adder

R. W. DORAN

Abstract—**An improved variation on the carry look-ahead adder has been proposed by Ling. Ling's approach is based on the propagation of a composite term in place of the conventional look-ahead carry. This approach gives an adder that is faster and less expensive.**

In this paper, Ling's adder is introduced and described in a general manner in order to expose the essence of his approach. From this reformulation, it is shown that there are many such variations on the carry look-ahead adder, a few of which share the desirable properties of Ling's adder.

Index Terms—**Binary adders, carry look-ahead adder, carry propagation, high-speed addition.**

I. Introduction

In a recent paper [1], Ling introduced a surprising variation of the conventional carry look-ahead adder, his adder being significantly better in cost and performance. Ling's exposition is based on a detailed case analysis of the behavior of the adder. In order to explain the concept behind Ling's work, we first derive his adder in a more general manner and discuss its advantages. By generalizing the approach, we will see that there are many other possible variations on the adder. We will explore these a little and show that there are other variations that share the advantages of Ling's adder.

II. Ling's Adder

In a conventional adder [2], to add the two numbers

$$A = a_0 2^n,\ a_1 2^{n-1},\ \cdots,\ a_n 2^0$$

and

$$B = b_0 2^n,\ b_1 2^{n-1},\ \cdots,\ b_n 2^0$$

we first form the local carry generate and propagate terms[1]:

$$g_i = a_i b_i$$

$$p_i = a_i \oplus b_i.$$

Then, with a ripple or tree circuit, we form the global "carry-out" terms resulting from the recurrence relation

$$G_i = g_i + p_i G_{i+1}. \tag{1}$$

Manuscript received February 13, 1986; revised November 13, 1986. This work was begun under the sponsorship of Amdahl Corporation and is released with their approval.

The author is with the Department of Computer Science, University of Auckland, Auckland, New Zealand.

IEEE Log Number 8716369.

[1] We will use + for "OR," ⊕ for "Exclusive OR," and proximity for "AND"—in order of increasing priority.

Finally, we form the sum S of A and B using the local expressions

$$S_i = p_i \oplus G_{i+1}. \tag{2}$$

In the conventional adder, the terms G_i have, as described, physical significance; however, an arbitrary function could be propagated, as long as the sum terms could then be derived. Ling's approach is to replace G_i with

$$H_i = G_i + G_{i+1} \tag{3}$$

i.e., H_i is true if "something interesting happens at bit i"—there is a carry out or a carry in. Before H_i can be propagated, it must first be expressed as a recurrence relation. Let us approach this by first deriving the inverse of (3), G_i expressed in terms of H_i. Consider the terms that comprise G_i in (1)

$g_i : g_i \supset G_i$ from (1) itself and

$G_i \supset H_i$ from (3) so, therefore,

$g_i \supset H_i$ and so

$$g_i \equiv g_i H_i \tag{4}$$

$$\begin{aligned} p_i G_{i+1} &:\equiv p_i G_{i+1} + p_i g_i + p_i G_{i+1} \text{ (as } p_i g_i = 0) \\ &\equiv p_i (G_{i+1} + G_i) \\ &\equiv p_i H_i. \end{aligned} \tag{5}$$

Substituting (4) and (5) in (1), we get

$$G_i = g_i H_i + p_i H_i$$

i.e.,

$$G_i = t_i H_i \tag{6}$$

where

$$t_i = a_i + b_i.$$

Now, as

$$\begin{aligned} H_i &= G_i + G_{i+1} \\ &= g_i + p_i G_{i+1} + G_{i+1} \\ &= g_i + G_{i+1}, \end{aligned}$$

we have from (6), Ling's recurrence relation

$$H_i = g_i + t_{i+1} H_{i+1}.$$

To determine the sum S, we first propagate H_i with circuits similar to

0018-9340/88/0900-1110$01.00

those used for G_i. Now substitute (6) in definition (2):

$$\begin{aligned}S_i &= p_i \oplus t_{i+1}H_{i+1}\\ &= \bar{p}_i t_{i+1}H_{i+1} + p_i\overline{(t_{i+1}H_{i+1})}\\ &= (g_i+\bar{t}_i)t_{i+1}H_{i+1} + t_i\bar{g}_i\overline{(t_{i+1}H_{i+1})}\\ &\quad (\text{as } \bar{p}_i = g_i+\bar{t}_i,\ p_i = t_i\bar{g}_i)\\ &= \bar{t}_i t_{i+1}H_{i+1} + t_i\bar{g}_i\overline{(t_{i+1}H_{i+1})} + g_i t_{i+1}H_{i+1}\\ &= \bar{t}_i(g_i + t_{i+1}H_{i+1}) + t_i\overline{(g_i+t_{i+1}H_{i+1})}\\ &\quad + g_i t_{i+1}H_{i+1} \qquad (\text{as } \bar{t}_i g_i = 0)\\ &= \bar{t}_i H_i + t_i\bar{H}_i + g_i t_{i+1}H_{i+1}\end{aligned}$$

i.e.,

$$S_i = t_i \oplus H_i + g_i t_{i+1}H_{i+1}$$

which is Ling's equation for completing the sum, the physical significance of which is not obvious!

In summary, the conventional adder implements

$$p_i = a_i \oplus b_i,\quad g_i = a_i b_i$$
$$G_i = g_i + p_i G_{i+1}$$
$$S_i = p_i \oplus G_{i+1}$$

whereas Ling's adder implements

$$t_i = a_i + b_i,\quad g_i = a_i b_i$$
$$H_i = g_i + t_{i+1}H_{i+1}$$
$$S_i = t_i \oplus H_i + g_i t_{i+1}H_{i+1}.$$

III. Advantage of Ling's Adder

On the face of it, Ling seems to have derived an adder with a simple start ($a_i + b_i$ versus $a_i \oplus b_i$) but a more complex conclusion. However, the most important difference lies in the recurrence relation.

Consider the conventional recurrence relation

$$G_i = g_i + p_i G_{i+1}.$$

The term p_i is usually more expensive to evaluate than the t_{i+1} term used by Ling. However, when that is the case, the conventional relation is modified as follows.

$$\begin{aligned}G_i &= g_i + g_i G_{i+1} + p_i G_{i+1}\\ &= g_i + (g_i + p_i)G_{i+1}\end{aligned}$$

i.e.,

$$G_i = g_i + t_i G_{i+1}.$$

Ling's relation has the form

$$H_i = g_i + t_{i+1}H_{i+1}$$

the sole difference thus being the index of t.

To see the significance of this difference, consider the expansions to four levels

$$G_0 = g_0 + t_0 g_1 + t_0 t_1 g_2 + t_0 t_1 t_2 g_3$$

whereas

$$\begin{aligned}H_0 &= g_0 + t_1 g_1 + t_1 t_2 g_2 + t_1 t_2 t_3 g_3\\ &= g_0 + g_1 + t_1 g_2 + t_1 t_2 g_3 \qquad (\text{as } g_i \supset t_i).\end{aligned}$$

The expansion of Ling's recurrence is considerably cheaper than the conventional: G_0 has ten inputs to four gates, the widest having four inputs, whereas H_0 has seven inputs to three gates, the widest having three inputs.

By considering similar cases, we can see that Ling's method is generally superior. Although the generation of the sum is more complex, this seems to be more than offset by the savings in carry propagation.

However, whether or not the designer can capitalize on the advantages depends on how well the method fits in with the available circuit elements. There may well be advantages in alternative formulations that more closely match the hardware. One is led to question whether there are other adders like Ling's that we can choose from.

IV. 32 Adders

There clearly are recurrence relations other than G_i and H_i, for example, $\bar{G}_i$ and $\bar{H}_i$. To find all recurrence relations that may be used, replace G_i with an arbitrary logical function of a_i, b_i, and G_{i+1}. Write this in a normalized form:

$$X_i = \psi(a_i, b_i)G_{i+1} + \phi(a_i, b_i)\bar{G}_{i+1}.$$

X_i must be symmetric in a_i and b_i, so ψ and ϕ must also be symmetric. But there are only eight symmetric functions in a_i and b_i, viz.,

0	$p_i = a_i\bar{b}_i + \bar{a}_i b_i$
$\bar{t}_i = \bar{a}_i\bar{b}_i$	$\bar{g}_i = p_i + \bar{t}_i$
$g_i = a_i b_i$	$t_i = p_i + g_i$
$\bar{p}_i = \bar{a}_i\bar{b}_i + a_i b_i$	$1 = p_i + \bar{p}_i.$

Thus, we can have at most 64 distinct functions X_i, but not all of these contain sufficient information to derive the sum locally ($X_i \equiv 0$ is a striking example).

If we can derive the sum locally from X_i, then we can certainly derive G_i and if we can derive G_i from X_i locally, we can immediately derive the sum. Thus, finding the sum locally from X_i is equivalent to deriving G_i from X_i locally.

Consider the functions $X_i = (p_i + x)G_{i+1} + (p_i + y)\bar{G}_{i+1}$ where $x, y \in \{0, \bar{t}_i, g_i, \bar{p}_i\}$, i.e., $\subset \bar{p}_i$.

If $p_i = 0$ then $G_i = g_i$ (from $G_i = g_i + p_i G_{i+1}$) and is derived locally, but if $p_i = 1$ then $x = 0$ and $y = 0$ so

$$X_i = G_{i+1} + \bar{G}_{i+1} = 1$$

and

$$G_i = g_i + p_i G_{i+1} = G_{i+1}$$

i.e., X_i is of no aid in determining G_i locally, so X_i does not form an adder. Similarly, $X_i = xG_{i+1} + y\bar{G}_{i+1}$ is of no use to us. That leaves us with

$$X_i = (p_i + x)G_{i+1} + y\bar{G}_{i+1}$$

and

$$X_i = xG_{i+1} + (p_i + y)\bar{G}_{i+1}$$

which, as we shall see can all form adders.

Consider the first case:

$$X_i=(p_i+x)G_{i+1}+y\bar{G}_{i+1}$$

as before, if $p_i = 0$ then $G_i = g_i$ but if $p_i = 1$ then

$$X_i=G_{i+1}$$

and

$$G_i=G_{i+1}=X_i.$$

Thus, G_i is determined from X_i, so X_i forms an adder.

Summarizing the above reasoning,

$$G_i=\bar{p}_ig_i+p_iX_i=g_i+p_iX_i$$

which may be expressed equivalently as

$$G_i=g_i+t_iX_i \text{ or } G_i=t_iX_i+g_i\bar{X}_i.$$

We can treat the second case similarly. Because there are four distinct values of x and y, we find that there are 32 adders in all, split into two classes. In summary,

$$X_i=(p_i+x)G_{i+1}+y\bar{G}_{i+1} \qquad X_i=xG_{i+1}+(p_i+y)\bar{G}_{i+1}$$

$$\bar{X}_i=\overline{(p_i+x)}G_{i+1}+\bar{y}\bar{G}_{i+1} \qquad \bar{X}_i=\bar{x}G_{i+1}+\overline{(p_i+y)}\bar{G}_{i+1}$$

$$\begin{aligned} G_i&=g_i+p_iX_i & G_i&=g_i+p_i\bar{X}_i\\ &=g_i+t_iX_i & &=g_i+t_i\bar{X}_i\\ &=t_iX_i+g_i\bar{X}_i & &=g_iX_i+t_i\bar{X}_i\\ \bar{G}_i&=\bar{t}_i+\bar{p}_i\bar{X}_i & \bar{G}_i&=\bar{t}_i+\bar{p}_iX_i\\ &=\bar{t}_i+\bar{g}_i\bar{X}_i & &=\bar{t}_i+\bar{g}_iX_i\\ &=\bar{t}_iX_i+\bar{g}_i\bar{X}_i & &=\bar{g}_iX_i+\bar{t}_i\bar{X}_i. \end{aligned}$$

There is not much to be said about these adders in general. Most are uninteresting, e.g.,

$$x=\bar{p}_i,\ y=0 \text{ in } X_i=(p_i+x)G_{i+1}+y\bar{G}_{i+1}$$

gives

$$X_i=G_{i+1}$$

and

$$x=g_i,\ y=g_i$$

gives

$$X_i=t_iG_{i+1}+g_i\bar{G}_{i+1},$$

i.e.,

$$X_i=g_i+t_iG_{i+1}=G_i.$$

However, some do have interesting equations, for example,

$$x=\bar{p}_i,\ y=0 \text{ in } X_i=xG_{i+1}+(p_i+y)\bar{G}_{i+1}$$

$$X_i=\bar{p}_iG_{i+1}+p_i\bar{G}_{i+1}=p_i\oplus G_{i+1}. \qquad (1)$$

Substituting for G_{i+1} and $\bar{G}_{i+1}$,

$$X_i=\bar{p}_i(g_{i+1}X_{i+1}+t_{i+1}\bar{X}_{i+1})+p_i(\bar{g}_{i+1}X_{i+1}+\bar{t}_{i+1}\bar{X}_{i+1})$$

i.e.,

$$X_i=(p_i\oplus g_{i+1})X_{i+1}+(p_i\oplus t_{i+1})\bar{X}_{i+1}. \qquad (2)$$

As $S_i = p_i \oplus G_{i+1}$, we can see that the recurrence relation (2) generates the sum *directly*. This example is certainly simple in concept but it does not appear to have much value when examined in detail. What we need to select are the adders with similar properties to Ling's adder.

V. Four Adders of Ling's Type

Ling's adder had two desirable properties in its recurrence relation

$$H_i=g_i+t_{i+1}H_{i+1}$$

- only H_{i+1} occurs, not $\bar{H}_{i+1}$
- the coefficient of H_{i+1} involves $i+1$th terms only.

Consider the first property as applied to our first general recurrence relation

$$X_i=(p_i+x)G_{i+1}+y\bar{G}_{i+1}$$

i.e.,

$$\begin{aligned} X_i&=(p_i+x)[t_{i+1}X_{i+1}+g_{i+1}\bar{X}_{i+1}]+y[\bar{t}_{i+1}X_{i+1}+\bar{g}_{i+1}\bar{X}_{i+1}]\\ &=[(p_i+x)t_{i+1}+y\bar{t}_{i+1}]X_{i+1}+[(p_i+x)g_{i+1}+y\bar{g}_{i+1}]\bar{X}_{i+1} \end{aligned}$$

here the term in $\bar{X}_{i+1}$ can be absorbed into X_i if

$$[(p_i+x)g_{i+1}+y\bar{g}_{i+1}] \subset [(p_i+x)t_{i+1}+y\bar{t}_{i+1}]$$

i.e., if

$$[(p_i+x)g_{i+1}+yp_{i+1}+y\bar{t}_{i+1}] \subset [(p_i+x)g_{i+1} + (p_i+x)p_{i+1}+y\bar{t}_{i+1}]$$

(as $\bar{g}_{i+1}=p_{i+1}+\bar{t}_{i+1}$ and $t_{i+1}=g_{i+1}+p_{i+1}$)

i.e., if

$$yp_{i+1} \subset (P_i+x)p_{i+1}$$

(as g_{i+1}, p_{i+1}, $\bar{t}_{i+1}$ are disjoint)

i.e., if

$$y \subset p_i+x$$

i.e., if

$$y \subset x \qquad (\text{as } y,\ x \subset \bar{p}_i).$$

This condition allows the definition of X_i to be reduced to

$$X_i=y+(p_i+x)G_{i+1}$$

and the recurrence relation to

$$X_i=(p_i+x)g_{i+1}+y+[(p_i+x)t_{i+1}+y]X_{i+1}.$$

The second desirable property arises when $x = \bar{p}_i$, for then

$$\begin{aligned} X_i&=g_{i+1}+y+(t_{i+1}+y)X_{i+1}\\ &=g_{i+1}+y+t_{i+1}X_{i+1}. \end{aligned}$$

In this case,

$$X_i=y+G_{i+1}.$$

A third nice property of Ling's adder is the simple relationship

$$G_i=t_iX_i$$

TABLE I
EQUATIONS DEFINING FOUR VARIANTS OF THE CARRY LOOK-AHEAD ADDER

X_i :	$G_{i+1} + g_i\bar{G}_{i+1}$	$G_{i+1} + \bar{p}_i\bar{G}_{i+1}$	$\bar{t}_iG_{i+1} + \bar{G}_{i+1}$	$\bar{p}_iG_{i+1} + \bar{G}_{i+1}$
X_i :	$g_i + G_{i+1}$	$\bar{p}_i + G_{i+1}$	$\bar{t} + \bar{G}_{i+1}$	$\bar{p}_i + \bar{G}_{i+1}$
G_i :	t_iX_i	t_iX_i	$g_i + \bar{X}_i$	$g_i + \bar{X}_i$
$\bar{G}_i$:	$\bar{t}_i + \bar{X}_i$	$\bar{t}_i + \bar{X}_i$	$\bar{g}_iX_i$	$\bar{g}_iX_i$
X_i :	$g_i + t_{i+1}X_{i+1}$	$\bar{p}_i + t_{i+1}X_{i+1}$	$\bar{t}_i + \bar{g}_{i+1}X_{i+1}$	$\bar{p}_i + \bar{g}_{i+1}X_{i+1}$
$\bar{X}_i$:	$\bar{g}_it_{i+1} + \bar{g}_i\bar{X}_{i+1}$	$p_i\bar{t}_{i+1} + p_i\bar{X}_{i+1}$	$t_ig_{i+1} + t_i\bar{X}_{i+1}$	$p_ig_{i+1} + p_i\bar{X}_{i+1}$
S_i :	$t_i \oplus X_i + g_it_{i+1}X_{i+1}$	$\bar{X}_i + \bar{p}_it_{i+1}X_{i+1}$	$\overline{\bar{g}_i \oplus X_i + \bar{t}_i\bar{g}_{i+1}X_{i+1}}$	$\overline{\bar{X}_i + \bar{p}_i\bar{g}_{i+1}X_{i+1}}$

i.e.,

$$G_i = yt_i + t_iG_{i+1}.$$

This occurs only if

$$g_i = yt_i \qquad (\text{as } G_i = g_i + t_iG_{i+1})$$

i.e., if

$$y = \bar{p}_i \text{ or } y = g_i.$$

With this additional property,

$$X_i = y + t_{i+1}X_{i+1}.$$

In summary, X_i has all three nice properties of Ling's adder if $x = \overline{p_i}$, and if $y = \overline{p_i}$ or $y = g_i$. Ling's adder corresponds to $y = g_i$.

Consider the other case: $x = \bar{p}_i$, $y = \overline{p_i}$

$$X_i = \bar{p}_i + G_{i+1}$$

$$G_i = t_iX_i$$

$$X_i = \bar{p}_i + t_{i+1}X_{i+1}$$

$$\begin{aligned} S_i &= p_i \oplus G_{i+1} \\ &= p_i\bar{G}_{i+1} + \overline{p_i}G_{i+1} \\ &= \overline{\bar{p}_i + G_{i+1}} + \bar{p}_it_{i+1}X_{i+1} \\ &= \bar{X}_i + \bar{p}_it_{i+1}X_{i+1}. \end{aligned}$$

For the other set of adders $X_i = xG_{i+1} + (p_i + y)\bar{G}_{i+1}$, we can find similarly that Ling's properties are enjoyed when $y = p_i$ and $x = \bar{p}_i$ or $\bar{t}_i$.

The four adders are as in Table I. Note that the last two adders find the inverse of the sum more directly than the sum itself.

ACKNOWLEDGMENT

Thanks are due to colleagues at Amdahl for their comments (in particular to R. J. Bishop, L. I. Dickman, and Dr. S. Lee) and to Dr. P. B. Gibbons at Auckland University for a stalwart job of checking.

REFERENCES

[1] H. Ling, "High speed binary adder," *IBM J. Res. Develop.*, vol. 25, p. 156, May 1981.

[2] See, for example, N. R. Scott, *Computer Number Systems and Arithmetic*. Englewood Cliffs, NJ: Prentice-Hall, 1985.

Efficient Testing of Optimal Time Adders

BERND BECKER

Abstract—**We consider the design of two well-known optimal time adders: the "carry look-ahead" adder [6] and the "conditional sum" adder [13].**

It is shown that 6 $\log_2(n)$ − 4 and 6 $\log_2(n)$ + 2 test patterns suffice to completely test the n-bit carry look-ahead adder and the n-bit conditional sum adder with respect to the single stuck-at fault model (for a given set of basic cells).

Index Terms—**Arithmetic circuits, complete test set, detectable fault, logical design, regular structure, single stuck-at fault model, test pattern, testing, VLSI chip.**

I. INTRODUCTION

The establishment of the correct behavior of a given VLSI chip is a problem which gains renewed importance and attention for the production process by the following fact: with increasing chip complexity, automatic test pattern generation (based on the D-algorithm or a "related" algorithm) is becoming very costly or even computationally infeasible in the general case. (The computation time may be exponential in the size of the circuit!) Therefore, it is useful to develop specific methods for important classes of circuits, e.g., circuits with regular structures such as PLA's, memories, or arithmetical units.

Work in this direction has been done by several authors (see, e.g., [14], [8], [1], [11], [10], [9]). It turns out that in many cases, a

Manuscript received February 25, 1986; revised January 12, 1987. This work was supported by the DFG (Deutsche Forschungsgemeinschaft), SFB 124, TP B1, "VLSI-Entwurfsmethoden und Parallelität."

The author is with Angewandte Mathematik und Informatik, Universität des Saarlandes, D-6600 Saarbrücken, West Germany.

IEEE Log Number 8716365.