not required, i.e., only six tests are required. However, this reduction in the size of $T_1$ is not possible in general.

ACKNOWLEDGMENT

The author wishes to thank Prof. S. V. Rangaswamy and the referee for their suggestions to improve the presentation of this note.

REFERENCES


Comment on "Some New Results on Average Worst Case Carry"

C.K. YUEN

When reading the above paper, the unwary person might easily come to the conclusion that "most carries are short." Actually, such a statement needs to be qualified. In fact, if a computer engineer designs an adder on this basis, he would probably find that the average add time is not as short as the theoretical analysis would lead him to expect. This is because Briley's analysis, and that of his predecessors Burks, Goldstine, and von Neumann [1], are both dependent on the assumption that the numbers being added are evenly distributed among $2^n$ possible bit combinations. In actual computation, most of the integers tend to have small magnitudes. Thus, a negative integer, if expressed in complement form, tends to have a long string of leading ones. When two such numbers are added, or when one of these is added to a positive integer giving a positive sum, a long string of carries would be generated. This means that in real computing situations the average carry might be much longer than Briley's analysis would indicate.

To get some idea of the effect of uneven distribution, let us take a very crude picture. We assume a word length of $n$, but also assume that the integers actually occurring are limited to the range $-2^n + 1$ to $2^n - 1$. Moreover, we assume that the numbers have probability $p$ of being positive, but each positive bit combination is equally likely. Similarly, they have a chance of $1 - p$ of being negative, with equal probability for each negative bit combination within the specified range. During addition in 2's complement, four cases are possible.

Case 1: Both addends $+$, with chance $p^2$.

Case 2: One addend $+$, one $-$, result $+$, with chance $p(1 - p)$.

Case 3: One addend $+$, one $-$, result $-$, with chance $p(1 - p)$.

Case 4: Both addends $-$, with chance $(1 - p)^2$.

Cases 1 and 2 follow Briley's analysis, which gives an approximate average carry length of $\log_2 m - \frac{1}{2}$. In Cases 3 and 4, however, we have a carry string through $n - m$ leading digits, so the average carry length is at least $n - m$. Adding the four cases together, we get an approximate average carry length of

$$-p/2 + p \log_2 m + (1 - p)(n - m).$$

For $p = 0.5$, $n = 48$ and $m = 32$, we have a figure of 10.25, compared with Briley's result of 5.1. Even if we assume that positive integers occur much more often than negative ones, and take $p = 0.9$, we still get a figure of 5.65. If we only have positive integers, then $p = 1$ and the average carry length is around 4.5.

Thus, our crude analysis shows that, in complement addition, the effect of uneven distribution is to increase carry length. But if we use

signed addition-subtraction then the average carry-borrow length is, in fact, shorter than Briley's analysis indicates.

It should be interesting to obtain some empirical data on the distribution of integers during machine executions through runtime monitoring. However, it must be pointed out that such results would probably vary from machine to machine and from program to program. For example, in FORTRAN execution most integers would be array indices and do-loop parameters, likely to be positive and small. In programs with much address manipulations there would be absolute addresses, which tend to cluster around the locations where most data are stored, and address increments, which tend to be small. One must thus be cautious when making generalized statements based on one's empirical results. Intuitively, it appears likely that small positive integers occur most frequently, followed by negative integers near zero, with numbers having large magnitudes occurring very rarely. There are, for example, few programs which do not make frequent use of the three integers $0$, $+1$, and $-1$. It might be pointed out that addition of $-1$ to any nonzero number in complement form generates a carry of length $n$.

REFERENCES


Comments on "Multiple Fault Detection in Combinational Networks"

PRABHAKAR GOEL AND DANIEL P. SIEWIOREK

Abstract—Some comments on a recent contribution on multiple fault detection using test sets for single fault detection are presented. A counter example that shows some defects in generalizing from a tree to an arbitrary network are also included.

Index Terms—Combinational networks, fan-out, multiple faults, single fault detection set.

In the above correspondence1 an attempt is made to generalize results established for a uniform combinational tree network to a more general combinational network. In particular there is the following theorem.

Theorem 9: Given any network with independent inputs where $K$ is the maximum gate fan-in, $t$ is the number of levels in the maximum path length, then the number of faults is bounded by

$$|F| < K^t + K^{t-1}$$

and there exists a $T$ set such that

$$1 < |T| < K^d(K^t + 1)$$

$$l = 2q + r, \quad 0 < r < 2$$

Manuscript received November 15, 1972; revised August 15, 1973. P. Goel was with the Department of Electrical Engineering, Carnegie-Mellon University, Pittsburgh, Pa. He is now with the Systems Products Division, IBM Corporation, East Fishkill, N.Y.

D. Siewiorek is with the Department of Electrical Engineering and the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pa.